Soft Comput (2018) 22:4047–4070
https://doi.org/10.1007/s00500-017-2615-6

METHODOLOGIES AND APPLICATION

Convergence analysis of standard particle swarm optimization algorithm and its improvement

Weiyi Qian[1] · Ming Li[1]

Published online: 18 May 2017

Abstract Standard particle swarm optimization (PSO) algorithm is a kind of stochastic optimization algorithm. Its convergence, based on probability theory, is analyzed in detail. We prove that the standard PSO algorithm is convergence with probability 1 under certain condition. Then, a new improved particle swarm optimization (IPSO) algorithm is proposed to ensure that IPSO algorithm is convergence with probability 1. In order to balance the exploration and exploitation abilities of IPSO algorithm, we propose the exploration and exploitation operators and weight the two operators in IPSO algorithm. Finally, IPSO algorithm is tested on 13 benchmark test functions and compared with the other algorithms published in the recent literature. The numerical results confirm that IPSO algorithm has the better performance in solving nonlinear functions.

Keywords Standard particle swarm optimization · Analysis of algorithm · Convergence in probability · Global optimization

1 Introduction

Particle swarm optimization (PSO) was introduced by Kennedy and Eberhart in 1995 for solving optimization problems (Kennedy and Eberhart 1995). Currently, it is a computationally effective and widely used stochastic optimization algorithm. From the time of initial introduction of the PSO algorithm, various improved PSO algorithms have been proposed in order to enhance its performance (Ding et al. 2014; Lim and Isa 2014; Wang et al. 2014; Yu and Zhang 2014; Zou et al. 2014; Shi and Eberhart 1999; Nickabadi et al. 2011; Feng et al. 2007; Shen et al. 2010; Adewumi and Arasomwan 2016; Chuang et al. 2011; Gandomi et al. 2013; Arasomwan and Adewumi 2014; Malik et al. 2007; Ali and Kaelo 2008), and it has been applied in various areas (Behnamian and Fatemi 2010; Afshar 2012; Victoire and Jeyakumar 2005; Franken and Engelbrecht 2005). Compared with the improvement and application researches of PSO algorithm, its theoretical analyses were relatively few and not very much perfect. The theoretical researches of PSO can be roughly classified into three categories, namely (a) the research on particle's trajectory; (b) the research on the parameter selection and stability; and (c) the research on convergence based on the framework of stochastic optimization algorithm.

The research on particle's trajectory: The first theoretical analysis of a simple model of PSO has been proposed by Ozcan and Mohan (1998, 1999), they analyzed the particle's trajectory of the simple model of PSO in one-dimensional and multidimensional space under the assumption that the random variable, the optimal position of individual particle, and the swarm's optimal position are constant. Clerc and Kennedy (2002) have investigated the characteristics and convergence of a particle trajectory based on the model presented in Ozcan and Mohan (1999). van den Bergh (2002) has analyzed the particle trajectory for the model of Ozcan and Mohan (1999) with inertia weight. Li et al. (2006) have made a thorough research on the stability of particle's trajectory of PSO based on difference equation and Z transform.

The research on the parameter selection and stability: The research on the parameter selection and stability of PSO is

Communicated by V. Loia.

✉ Weiyi Qian
weiyiqian2012@sina.cn

[1] College of Mathematics and Physics, Bohai University, Jinzhou 121000, China

mainly based on the dynamic system stability theory. Trelea has analyzed convergence of PSO and given a set of parameter selection to balance the exploration and exploitation abilities of PSO (Trelea 2003). Fernandez-Martinez and Garcia-Gonzalo have investigated the stability of the discrete PSO and continue PSO (Fernandez-Martinez et al. 2011). Kadirkamanathan et al. (2006) have described the optimal particle as a nonlinear feedback system when PSO stagnated, and analyzed the sufficient condition of stability based on Lyapunov theorem. Emara and Fattah (2004) have proposed a continue model of PSO and proved the stability of the model based on Lyapunov theorem. Rapaic and Kanovic (2009) have presented a formal convergence analysis of PSO with time-varying parameters.

The research on convergence based on the framework of stochastic optimization algorithm: The above analyses did not consider the randomness of the PSO algorithm. Obviously, these results more or less deviate from the real particle swarm system due to the loss of the randomness (Jiang et al. 2007). Therefore, some theoretical analyses of PSO have been presented under random environment. For standard particle swarm optimization algorithm, Jiang et al. (2007) have analyzed the convergent condition of variance and expectation of the particle's position based on stochastic process and proposed the parameter selection guidelines by convergence analysis. van den Bergh and Engelbrecht (2006) have proved that standard particle swarm optimization algorithm was neither local convergent nor global convergence by using the convergence criterion proposed by Solis and Wets and proposed an improved PSO algorithm to ensuring the local convergence. Chen and Chen (2011) have introduced a statistical interpretation of PSO and given theoretical results on the convergence time by using the statistical model. Jin et al. (2007) have presented a sufficient condition of the system mean-square to be stable based on stochastic process.

For a deterministic optimization algorithm, convergence analysis means that the iterative sequence generated by the algorithm converges to the optimal solution. Similarly, the convergence of a stochastic optimization algorithm should be that the iterative sequence generated by the algorithm converges to the optimal solution with probability 1. Aforementioned theoretical analyzes were mainly based on the dynamic system theory, difference equation theory, and stochastic process theory and did not use the Markov chain theory to consider the probability convergence of PSO. Based on the Markov chain theory, Pan et al. (2013) have researched the probability convergence of the standard PSO algorithm. But the transition probability of a particle was calculated by intuitive geometric method without considering the density of a probability distribution, the correctness of result can be influenced due to the loss of rigorous mathematical proof. It is necessary that the probability convergence of the standard PSO algorithm is proved by rigorous mathematical methods and the improved PSO algorithm that is convergence with probability 1 is proposed.

In this paper, we derive the transition probability of a particle by using mathematical method for the standard PSO algorithm. Based on the derived result, we prove that the standard PSO algorithm is convergence with probability 1 under certain condition. Then, we proposed a new improved particle swarm optimization algorithm which is not only convergence with probability 1, but also improve convergence rate and the solution quality.

This paper is organized as follows. In Sect. 2, the standard PSO algorithm is described in detail. Section 3 calculates the density function of update particle. In Sect. 4, the convergence analysis is made. Proposed algorithm and numerical simulations are given in Sect. 5. Finally, Sect. 6 gives conclusions.

2 The standard PSO algorithm

In standard PSO algorithm, each solution in the search region $S = \{x \in R^D | l_i \leq x_i \leq u_i, l_i, u_i \in R, \quad i = 1, 2, \ldots, D\}$ can be modeled as a particle which moves in S by the velocity of the particle. Each particle has position variable and velocity variable. Suppose that a swarm consists of N_p particles, the position and velocity of the ith particle of the swarm are denoted as a D-dimensional vectors $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})^{\mathrm{T}} \in S$ and $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD})^{\mathrm{T}} \in V = \{(v_1, v_2, \ldots, v_D) | v_{\min} \leq v_i \leq v_{\max}\}$, $i = 1, 2, \ldots, N_p$, respectively. The previous best position of the ith particle is represented by $P_i = (p_{i1}, p_{i2}, \ldots, p_{iD})^{\mathrm{T}}$. The previous best position of the swarm is denoted as $P_g = (p_{g1}, p_{g2}, \ldots, p_{gD})^{\mathrm{T}}$. At iteration k, the d th dimension of particle i's velocity and position are updated as follows:

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

$$x_{id}^{k+1} = x_{id}^k + v_{id}^{k+1}, \tag{2}$$

where c_1 and c_2 are two acceleration factors, r_1 and r_2 are two random numbers, uniformly distributed in the range [0, 1], and ω is inertia factor which can defined as follows:

$$\omega_k = (\omega_{\text{start}} - \omega_{\text{end}}) \left(\frac{K_{\max} - k}{K_{\max}}\right) + \omega_{\text{end}}, \tag{3}$$

where ω_{start} and ω_{end} are the initial and final values of inertia weight, k is the current iteration number, $K_{\max}$ is the maximum iteration number.

According to (1) and (2), the position update equation can also be described as

$$x_{id}^{k+1} = x_{id}^{k} + \omega_k v_{id}^{k} + c_1 r_1 \left(p_{id}^{k} - x_{id}^{k}\right) + c_2 r_2 \left(p_{gd}^{k} - x_{id}^{k}\right). \tag{4}$$

3 Probability density function

Let ξ and η be two independent uniform random variables distributed in [0, 1], we discuss the probability density function of the random variable $\zeta = a\xi + b\eta + c$, where a, b, and c are real numbers. Lemma 1 was introduced in the textbook of probability theory.

Lemma 1 *Assume that* (ξ, η) *is a two-dimensional continuous random variable, and it's joint density function is* $p(x, y)$. *If functions* $\begin{cases} u = g_1(x, y) \\ v = g_2(x, y) \end{cases}$ *have continuous partial derivative, and it exists uniquely inverse functions* $\begin{cases} x = x(u, v) \\ y = y(u, v). \end{cases}$ *Let* $\begin{cases} U = g_1(\xi, \eta) \\ V = g_2(\xi, \eta), \end{cases}$ *then the joint density function of* (U, V) *is defined as follows:*

$$p(u, v) = p(x(u, v), y(u, v))|J| \tag{5}$$

where $J = \frac{\partial(x,y)}{\partial(u,v)}$ *is Jacobian determinant.*

Lemma 2 *Assume that* ξ *and* η *are two mutually independent random variables, and uniformly distributed in the range* [0, 1], *then the probability density function of the random variable* $\zeta = a\xi + b\eta + c \, (a \neq 0, b \neq 0, a \neq b)$ *is*

$$p_\zeta(u) = \begin{cases} \frac{u-\alpha}{|a||b|}, & \text{if } \alpha \le u \le \beta \\ \frac{\min\{|a|,|b|\}}{|a||b|}, & \text{if } \beta \le u \le \gamma \\ \frac{\tau-u}{|a||b|}, & \text{if } \gamma \le u \le \tau \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $\alpha = \min\{0, a\} + \min\{0, b\} + c$, $\beta = \min\left\{\min\{0, a\} + \max\{0, b\}, \max\{0, a\} + \min\{0, b\}\right\} + c$, $\gamma = \max\{\min\{0, a\} + \max\{0, b\}, \max\{0, a\} + \min\{0, b\}\} + c$, *and* $\tau = \max\{0, a\} + \max\{0, b\} + c$.

Proof Since ξ and η are uniformly distributed random variables in [0, 1], the density functions of ξ and η are $p_\xi(x) = \begin{cases} 1, & \text{if } 0 \le x \le 1 \\ 0, & \text{otherwise} \end{cases}$ and $p_\eta(y) = \begin{cases} 1, & \text{if } 0 \le y \le 1 \\ 0, & \text{otherwise} \end{cases}$, respectively. ξ and η are mutually independent random variables; therefore, joint density function of (ξ, η) is $p(x, y) = p_\xi(x) \cdot p_\eta(y)$.

Let $\begin{cases} \zeta = a\xi + b\eta + c \\ \varsigma = a\xi \end{cases}$, then inverse functions of functions $\begin{cases} u = ax + by + c \\ v = ax \end{cases}$ are

$$\begin{cases} x = \frac{v}{a} \\ y = \frac{u-v-c}{b} \end{cases}. \tag{7}$$

Jacobian determinant of Eq. (7) is $J = -\frac{1}{ab}$. By Lemma 1, we know that the joint density function of (ζ, ς) is

$$\begin{aligned} p(u, v) &= p\left(\frac{v}{a}, \frac{u-v-c}{b}\right) \cdot |J| \\ &= p_\xi\left(\frac{v}{a}\right) p_\eta\left(\frac{u-v-c}{b}\right) \cdot \frac{1}{|ab|}. \end{aligned} \tag{8}$$

Thus, the density function of $\zeta = a\xi + b\eta + c$ is

$$p_\zeta(u) = \int_{-\infty}^{+\infty} p_\xi\left(\frac{v}{a}\right) p_\eta\left(\frac{u-v-c}{b}\right) \cdot \frac{1}{|ab|} \mathrm{d}v. \tag{9}$$

When $a, b > 0$, $p_\xi(\frac{v}{a}) \cdot p_\eta(\frac{u-v-c}{b}) = 1$ only and only if $0 \le v \le a$, $u - b - c \le v \le u - c$ and $c \le u \le a + b + c$. Hence, if $c \le u \le a + b + c$, then

$$\begin{aligned} p_\zeta(u) &= \int_{\max\{0,u-b-c\}}^{\min\{a,u-c\}} \frac{1}{ab} \mathrm{d}v \\ &= \frac{\min\{a, u-c\} - \max\{0, u-b-c\}}{ab} \\ &= \frac{a + b - |u-a-c| - |u-b-c|}{2ab}. \end{aligned}$$

If $u < c$ or $u > a + b + c$, then

$$p_\zeta(u) = 0.$$

This implies that

$$p_\zeta(u) = \begin{cases} \frac{a+b-|u-(a+c)|-|u-(b+c)|}{2ab}, & \text{if } c \le u \le a+b+c \\ 0, & \text{otherwise} \end{cases}. \tag{10}$$

From Eq. (10), it follows that when $a > b > 0$,

$$p_\zeta(u) = \begin{cases} \frac{u-c}{|a||b|}, & \text{if } c \le u \le b+c \\ \frac{b}{|a||b|}, & \text{if } b+c < u \le a+c \\ \frac{a+b+c-u}{|a||b|}, & \text{if } a+c < u \le a+b+c \\ 0, & \text{otherwise} \end{cases}, \tag{11}$$

and when $b > a > 0$,

$$p_\zeta(u) = \begin{cases} \frac{u-c}{|a||b|}, & \text{if } c \le u \le a+c \\ \frac{a}{|a||b|}, & \text{if } a+c < u \le b+c \\ \frac{a+b+c-u}{|a||b|}, & \text{if } b+c < u \le a+b+c \\ 0, & \text{otherwise} \end{cases}. \tag{12}$$

Similarly, when $a > 0, b < 0$ and $a > -b$, we can obtain

$$p_\zeta(u) = \begin{cases} \frac{u-(b+c)}{|a||b|}, & \text{if } b+c \le u \le c \\ \frac{-b}{|a||b|}, & \text{if } c \le u \le a+b+c \\ \frac{a+c-u}{|a||b|}, & \text{if } a+b+c \le u \le a+c \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

When $a > 0, b < 0$ and $a < -b$, we have

$$p_\zeta(u) = \begin{cases} \frac{u-(b+c)}{|a||b|}, & \text{if } b+c \le u \le a+b+c \\ \frac{a}{|a||b|}, & \text{if } a+b+c \le u \le c \\ \frac{a+c-u}{|a||b|}, & \text{if } c \le u \le a+c \\ 0, & \text{otherwise} \end{cases} . \tag{14}$$

When $a < 0, b > 0$ and $b > -a$, we have

$$p_\zeta(u) = \begin{cases} \frac{u-(a+c)}{|a||b|}, & \text{if } a+c \le u \le c \\ \frac{-a}{|a||b|}, & \text{if } c \le u \le a+b+c \\ \frac{b+c-u}{|a||b|}, & \text{if } a+b+c \le u \le b+c \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

When $a < 0, b > 0$ and $b < -a$, we get

$$p_\zeta(u) = \begin{cases} \frac{u-(a+c)}{|a||b|}, & \text{if } a+c \le u \le a+b+c \\ \frac{b}{|a||b|}, & \text{if } a+b+c \le u \le c \\ \frac{b+c-u}{|a||b|}, & \text{if } c \le u \le b+c \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

When $a < 0, b < 0$ and $a < b$, we get

$$p_\zeta(u) = \begin{cases} \frac{u-(a+b+c)}{|a||b|}, & \text{if } a+b+c \le u \le a+c \\ \frac{-b}{|a||b|}, & \text{if } a+c \le u \le b+c \\ \frac{c-u}{|a||b|}, & \text{if } b+c \le u \le c \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

When $a < 0, b < 0$ and $a > b$, we obtain

$$p_\zeta(u) = \begin{cases} \frac{u-(a+b+c)}{|a||b|}, & \text{if } a+b+c \le u \le b+c \\ \frac{-a}{|a||b|}, & \text{if } b+c \le u \le a+c \\ \frac{c-u}{|a||b|}, & \text{if } a+c \le u \le c \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

Let $\alpha = \min\{0, a\} + \min\{0, b\} + c$, $\beta = \min\left\{\min\{0, a\} + \max\{0, b\}, \max\{0, a\} + \min\{0, b\}\right\} + c$, $\gamma = \max\{\min\{0, a\} + \max\{0, b\}, \max\{0, a\} + \min\{0, b\}\} + c$, and $\tau = \max\{0, a\} + \max\{0, b\} + c$. Then, the unified form of Eqs. (11)–(18) can be written as follows:

$$p_\zeta(u) = \begin{cases} \frac{u-\alpha}{|a||b|}, & \text{if } \alpha \le u \le \beta \\ \frac{\min\{|a|,|b|\}}{|a||b|}, & \text{if } \beta \le u \le \gamma \\ \frac{\tau-u}{|a||b|}, & \text{if } \gamma \le u \le \tau \\ 0, & \text{otherwise} \end{cases} .$$

This completes the proof of the lemma.

If $a = b \neq 0$, then $\beta = \gamma$. By Lemma 2, the density function of $\zeta = a\xi + b\eta + c$ is

$$p_\zeta(u) = \begin{cases} \frac{u-\alpha}{|a||b|}, & \text{if } \alpha \le u \le \beta \\ \frac{\tau-u}{|a||b|}, & \text{if } \beta \le u \le \tau \\ 0, & \text{otherwise} \end{cases} . \tag{19}$$

If $a = 0$ and $b \neq 0$, then $\zeta = b\eta + c$ is one-dimensional random variable function. It is easy to get the density function of ζ

$$p_\zeta(u) = \begin{cases} \frac{1}{|b|}, & \text{if } \alpha \le u \le \tau \\ 0, & \text{otherwise} \end{cases} . \tag{20}$$

Similarly, If $a \neq 0$ and $b = 0$, we can obtain the density function of $\zeta = a\xi + c$

$$p_\zeta(u) = \begin{cases} \frac{1}{|a|}, & \text{if} \alpha \le u \le \tau \\ 0, & \text{otherwise} \end{cases} . \tag{21}$$

□

4 Convergence analysis

4.1 Definitions and notation

Definition 1 Let $f(x)$ is the real function, and $x^* \in S$, if

$$f(x^*) \le f(x), \quad \forall x \in S,$$

then x^* is said to be a global optimal solution on S.

Definition 2 $\forall \varepsilon > 0$, let

$$B_\varepsilon = \left\{x \in S \left| |f(x) - f(x^*)| < \varepsilon \right| \right\},$$

then B_ε is called a set of ε-optimal solutions where $x \in B_\varepsilon$ is said to be ε-optimal solution.

By Eq.(3), we know that the ith particle state in PSO swarm is determined by the vector $\varphi_i = (X_i, V_i, p_i, p_g)^{\mathrm{T}} \in \Gamma$, where $\Gamma = S \times V \times S \times S \in R^{4D}$.

Definition 3 Let

$$\Phi_{Np} = \left\{\Phi | \Phi = (\varphi_1, \varphi_2, \ldots, \varphi_{Np}), \varphi_i \in \Gamma, \ i = 1, 2, \ldots, Np\right\},$$

then $\varPhi_{Np}$ is said to be the state space, where $\varPhi \in \varPhi_{Np}$ is called a state.

Definition 4 Let

$$\varOmega = \left\{\omega | \omega = (\varPhi_1, \varPhi_2, \varPhi_3, \ldots), \varPhi_k \in \varPhi_{Np}, \forall k\right\},$$

then $\varOmega$ is said to be the sample space, $\omega \in \varOmega$ represents a sequence of states. In here $\varPhi_k = (\varphi_1^k, \varphi_2^k, \ldots, \varphi_{Np}^k)$ and $\varphi_i^k = (X_i^k, v_i^k, p_i^k, p_g^k)$, $i = 1, 2, \ldots, NP$.

We define the function,

$$\chi_B(\varPhi) = \sum_{i=1}^{Np} (1_B(X_i) + 1_B(p_i) + 1_B(p_g)), \quad \forall \varPhi \in \varPhi_{Np} \tag{22}$$

where $1_B(x)$ is the indication function of B, i.e., $1_B(x) = 1$ if $x \in B$, $1_B(x) = 0$ otherwise.

We define the transition probability,

$$p\{\chi_B\left(\varPhi_{k+1}\right) \neq 0\} = p\{\chi_B\left(\varPhi_{k+1}\right) \neq 0 | \varPhi_k = \varPhi\}, \quad \forall \varPhi \in \varPhi_{Np} \tag{23}$$

which is the conditional probability.

In this paper, assume that $f(x)$ is continuous function.

By the assumption, we know that there exists a positive real number δ such that

$$\mathrm{E}_\delta = \{(x_1, x_2, \ldots, x_D) | |x_i - x_i^*| < \delta, i{=}1, 2, \ldots, D\} \subset B_\varepsilon, \tag{24}$$

where x_i^* is the ith dimension of x^*. Notice that if $x_i^* = l_i$ or $x_i^* = u_i$, then the inequality $|x_i - x_i^*| < \delta$ is replaced by $0 \leq x_i - x_i^* < \delta$ or $\delta < x_i - x_i^* \leq 0$. For convenience, we only consider the case of Eq. (24) in this paper.

By Eq. (24), we know that $\mu(B_\varepsilon) > 0$ where μ is the Lebesgue measure on R^D.

4.2 Convergence proof

In standard PSO algorithm, we can see that the $(k + 1)$ th moment state $\varPhi_{k+1}$ of the particle swarm depends only on the kth moment state $\varPhi_k$. Therefore, the stochastic state sequence $\{\varPhi_k\}$ generated by the standard PSO algorithm is a Markov chain.

Lemma 3 *$\forall \varPhi \in \varPhi_{Np}$, $\chi_{B_\varepsilon}(\varPhi) \neq 0$ if and only if $p_g \in B_\varepsilon$.*

Proof Assume $p_g \notin B_\varepsilon$, then $1_{B_\varepsilon}(p_g) = 0$. p_g is the best position of the swarm, therefore $f(X_i) \geq f(p_g)$ and $f(p_i) \geq f(p_g)$, $\forall i$. Thus $x_i \notin B_\varepsilon$ and $p_i \notin B_\varepsilon$ which implies $1_{B_\varepsilon}(x_i) = 0$ and $1_{B_\varepsilon}(p_i) = 0$, $\forall i$. From (22), we have $\chi_{B_\varepsilon}(\varPhi) = 0$ which contradicts with $\chi_{B_\varepsilon}(\varPhi) \neq 0$.

Conversely, if $p_g \in B_\varepsilon$, then $1_{B_\varepsilon}(p_g) = 1$. By (22), $\chi_{B_\varepsilon}(\varPhi) \neq 0$. This completes the proof of the lemma. □

Lemma 4 *$\forall (\varPhi_1, \varPhi_2, \ldots) \in \varOmega$, if $\chi_{B_\varepsilon}(\varPhi_k) \neq 0$, then $\chi_{B_\varepsilon}\left(\varPhi_{k+1}\right) \neq 0$.*

Proof Since $\chi_{B_\varepsilon}(\varPhi_k) \neq 0$, by Lemma 3, we have $p_g^k \in B_\varepsilon$. By standard PSO algorithm, we know $f(p_g^{k+1}) \leq f\left(P_g^k\right)$which implies $p_g^{k+1} \in B_\varepsilon$. From Lemma 3, we get $\chi_{B_\varepsilon}\left(\varPhi_{k+1}\right) \neq 0$. This completes the proof of the lemma.

Let $a_{id}^k = c_1(p_{id}^k - x_{id}^k)$, $b_{id}^k = c_2(p_{gd}^k - x_{id}^k)$ and $c_{id}^k = x_{id}^k + \omega v_{id}^k$, then Eq. (4) can be described as follows

$$x_{id}^{k+1} = a_{id}^k r_1 + b_{id}^k r_2 + c_{id}^k \tag{25}$$

which is a random variable.

If a, b and c in Lemma 2 are replaced by a_{id}^k, b_{id}^k and c_{id}^k, respectively, then α, β, γ and τ are replaced by $\alpha_{id}^k, \beta_{id}^k, \gamma_{id}^k$ and τ_{id}^k, accordingly. □

Lemma 5 *$\forall \varPhi \in \varPhi_{Np}$, if $\varPhi_k = \varPhi$ and $x^* = (x_1^*, x_2^*, \ldots x_D^*) \in M^k = \bigcup_{i=1}^{Np} M_i^k$, $\forall k \geq 1$, then there exists a constant $\rho > 0$, such that*

$$p(x_{B_\varepsilon}\left(\varPhi_{k+1}\right) \neq 0) \geq \rho. \tag{26}$$

where $M_i^k = [\alpha_{i1}^k, \tau_{i1}^k] \times [\alpha_{i2}^k, \tau_{i2}^k] \times \cdots \times [\alpha_{iD}^k, \tau_{iD}^k]$, $i = 1, 2, \ldots, Np$, $k = 1, 2, 3, \ldots$ is a hypercube and $\varPhi_{k+1}$ is generated by the standard PSO algorithm.

Proof Sine $x^* = (x_1^*, x_2^*, \ldots x_D^*) \in M^k$, there exists at least $l \in \{1, 2, \ldots, Np\}$, such that $x^* \in M_l^k$. This can imply $x_d^* \in \lfloor \alpha_{ld}^k, \tau_{ld}^k \rfloor$, $d = 1, 2, \ldots, D$.

When $a_{ld}^k \neq 0, b_{ld}^k \neq 0$ and $a_{ld}^k \neq b_{ld}^k$, we consider the transition probability $p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right)$ from following two cases.

Case 1 Suppose that $x_d^* - \delta \geq \alpha_{ld}^k$ or $x_d^* + \delta \leq \tau_{ld}^k$.

By reason of $\alpha_{ld}^k \leq x_d^* \leq \tau_{ld}^k$, we have $\left|\left[\alpha_{ld}^k, \tau_{ld}^k\right] \cap \left[x_d^* {-} \delta, x_d^* {+} \delta\right]\right| {\geq} \delta$, where $|I|$ denotes the length of the interval I. Then, there exists at least one among $\left|\left[\alpha_{ld}^k, \beta_{ld}^k\right] \cap \left[x_d^* - \delta, x_d^* + \delta\right]\right|$, $\left|\left[\beta_{ld}^k, \gamma_{ld}^k\right] \cap \left[x_d^* - \delta, x_d^* + \delta\right]\right|$, and $\left|\left[\gamma_{ld}^k, \tau_{ld}^k\right] \cap \left[x_d^* - \delta, x_d^* + \delta\right]\right|$ which is equal or greater than $\frac{\delta}{3}$. If $\left|\left[\alpha_{ld}^k, \beta_{ld}^k\right] \cap \left[x_d^* - \delta, x_d^* + \delta\right]\right| \geq \frac{\delta}{3}$, then $\left|\left[\max\{\alpha_{ld}^k, x_d^* - \delta\}, \min\{\beta_{ld}^k, x_d^* + \delta\}\right]\right| \geq \frac{\delta}{3}$. By Lemma 2, we have

$$p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) = \int_{x_d^* - \delta}^{x_d^* + \delta} p_{x_{ld}^{k+1}}(u)\mathrm{d}u$$
$$\geq \int_{\max[\alpha_{ld}^k, x_d^* - \delta]}^{\min[\beta_{ld}^k, x_d^* + \delta]} \frac{u - \alpha_{ld}^k}{\left|a_{id}^k\right| \left|b_{id}^k\right|} \mathrm{d}u.$$

Let $\theta_1 = \max\left[\alpha_{ld}^k, x_d^* - \delta\right]$, $\theta_2 = \min\left[\beta_{ld}^k, x_d^* + \delta\right]$, then $\theta_2 - \theta_1 \geq \frac{\delta}{3}$, $\theta_1 - \alpha_{ld}^k \geq 0$. Thus, we have

$$\begin{aligned} & p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) \\ & \quad \geq \frac{1}{2\left|a_{id}^k\right|\left|b_{id}^k\right|}\left[\left(\theta_2 - \alpha_{ld}^k\right)^2 - \left(\theta_1 - \alpha_{ld}^k\right)^2\right] \\ & \quad = \frac{1}{2\left|a_{id}^k\right|\left|b_{id}^k\right|}\left(\theta_2 + \theta_1 - 2\alpha_{ld}^k\right)(\theta_2 - \theta_1) \\ & \quad \geq \frac{1}{2\left|a_{id}^k\right|\left|b_{id}^k\right|}\left(\frac{\delta}{3} + 2\theta_1 - 2\alpha_{ld}^k\right)\frac{\delta}{3} \\ & \quad \geq \frac{1}{2\left|a_{id}^k\right|\left|b_{id}^k\right|}\left(\frac{\delta}{3}\right)^2. \end{aligned}$$

Let $m = \max\limits_{1 \leq i \leq D}\{|l_i|, |u_i|\} > 0$, then $|a_{id}^k| \leq 2c_1 m$ and $|b_{id}^k| \leq 2c_2 m$. Hence, one has

$$p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) \geq \frac{1}{8c_1c_2m^2} \cdot \left(\frac{\delta}{3}\right)^2. \tag{27}$$

Similarly, if $\left|\left[\gamma_{ld}^k, \tau_{ld}^k\right] \cap \left[x_d^* - \delta, x_d^* + \delta\right]\right| \geq \frac{\delta}{3}$, we can obtain

$$\begin{aligned} & p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) \\ & \quad \geq \int_{\max[\gamma_{ld}^k, x_d^* - \delta]}^{\min[\tau_{ld}^k, x_d^* + \delta]} \frac{\tau_{ld}^k - u}{\left|a_{id}^k\right|\left|b_{id}^k\right|}\mathrm{d}u \geq \frac{1}{8c_1c_2m^2} \cdot \left(\frac{\delta}{3}\right)^2. \end{aligned} \tag{28}$$

If $\left|\left[\beta_{ld}^k, \gamma_{ld}^k\right] \cap \left[x_d^* - \delta, x_d^* + \delta\right]\right| \geq \frac{\delta}{3}$, by Lemma 2, we have

$$\begin{aligned} & p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) \\ & \quad \geq \int_{\max[\beta_{ld}^k, x_d^* - \delta]}^{\min[\gamma_{ld}^k, x_d^* + \delta]} \frac{\min\lfloor\left|a_{id}^k\right|\left|b_{id}^k\right|\rfloor}{\left|a_{id}^k\right|\left|b_{id}^k\right|}\mathrm{d}u \geq \frac{\min\{c_1, c_2\}}{4c_1c_2m} \cdot \frac{\delta}{3}. \end{aligned} \tag{29}$$

Case 2 Suppose that $x_d^* - \delta < \alpha_{ld}^k$ and $x_d^* + \delta > \tau_{ld}^k$.
By Lemma 2, we have

$$p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) = \int_{\alpha_{ld}^k}^{\tau_{ld}^k} p_{x_{ld}^{k+1}}(u)\mathrm{d}u = 1. \tag{30}$$

Similar to the above proof, when $a_{ld}^k \neq 0, b_{ld}^k \neq 0$ and $a_{ld}^k = b_{ld}^k$, by Eq. (19), we can get

$$p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) \geq \frac{1}{8c_1c_2m^2} \cdot \left(\frac{\delta}{2}\right)^2. \tag{31}$$

When $a_{ld}^k \neq 0, b_{ld}^k = 0$, by Eq. (21), we can obtain

$$p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) \geq \frac{\delta}{\left|a_{id}^k\right|} \geq \frac{\delta}{2c_1m}. \tag{32}$$

When $a_{ld}^k = 0, b_{ld}^k \neq 0$, by Eq. (20), we have

$$p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) \geq \frac{\delta}{\left|b_{id}^k\right|} \geq \frac{\delta}{2c_2m}. \tag{33}$$

When $a_{ld}^k = 0, b_{ld}^k = 0$, we have $\alpha_{ld}^k = \tau_{ld}^k = c_{ld}^k$. By $x_d^* \in \lfloor\alpha_{ld}^k, \tau_{ld}^k\rfloor$, we have $x_{ld}^{k+1} = c_{ld}^k = x_d^*$ which is not a random variable. In this case, we define

$$p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) = 1. \tag{34}$$

Let $\rho_1 = \min\left\{\frac{1}{8c_1c_2m^2}\left(\frac{\delta}{3}\right)^2, \frac{\min\{c_1,c_2\}}{4c_1c_2m}\frac{\delta}{3}, \frac{1}{8c_1c_2m^2}\left(\frac{\delta}{2}\right)^2, \frac{\delta}{2c_1m}, \frac{\delta}{2c_2m}, 1\right\} > 0$, from Eqs. (27)–(34), we can get

$$p(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta) \geq \rho_1 > 0 \tag{35}$$

Because each component of x_l^{k+1} is independent, by (35), we obtain,

$$\begin{aligned} p(x_l^{k+1} \in B_\varepsilon) &\geq p(x_l^{k+1} \in E_\delta) \\ &= \prod_{d=1}^{D} p\left(x_d^* - \delta < x_{ld}^{k+1} < x_d^* + \delta\right) \geq \rho_1^D > 0. \end{aligned}$$

Thus

$$p(x_{B_\varepsilon}(\Phi_{k+1}) \neq 0) \geq p(x_l^{k+1} \in B_\varepsilon) > \rho_1^D.$$

Let $\rho = \rho_1^D > 0$, we have

$$p(x_{B_\varepsilon}(\Phi_{k+1}) \neq 0) \geq \rho > 0.$$

This completes the proof of lemma. □

Theorem 1 *If there exists k_0 such that $x_{B_\varepsilon}(\Phi_{k_0}) \neq 0$, then the sequence $\left\{p_g^k\right\}$, $\forall k \geq k_0$ generated by the standard PSO algorithm can obtain the ε-optimum solution, i.e.,*

$$p_g^k \in B_\varepsilon, \forall k \geq k_0.$$

Proof Due to $x_{B_\varepsilon}(\Phi_{k_0}) \neq 0$, from Lemma 4, one has $x_{B_\varepsilon}(\Phi_k) \neq 0$, $\forall k \geq k_0$. By Lemma 3, we have $p_g^k \in B_\varepsilon$. This completes the proof of the theorem.

Theorem 2 *Suppose that $\forall(\Phi_1, \Phi_2, \Phi_3, \ldots) \in \Omega$ is a sequence generated by standard PSO algorithm. If there exists a subsequence $\{\Phi_{k_j}\}$ such that the condition of*

Lemma 5 is satisfied, then the sequence $\{p_g^k\}$ converges to the ε-optimum solution with probability one, i.e.,

$$\lim_{k\to\infty} p\{p_g^k \in B_\varepsilon\} = 1.$$

Proof By Lemma 4 and the Markovian property of the process, one has

$$\begin{aligned}
&p\{\chi_{B_\varepsilon}(\Phi_{k_{j+1}}) = 0\}\\
&\quad = p\{\chi_{B_\varepsilon}(\Phi_1) = 0, \chi_{B_\varepsilon}(\Phi_2) = 0, \ldots, \chi_{B_\varepsilon}(\Phi_{k_{j+1}}) = 0\}\\
&\quad = p\{\chi_{B_\varepsilon}(\Phi_1) = 0\} \cdot \prod_{t=1}^{k_{j+1}-1} p\{\chi_{B_\varepsilon}(\Phi_{t+1}) = 0\}\\
&\quad \le \prod_{t=1}^{k_{j+1}-1} (p\{\chi_{B_\varepsilon}(\Phi_{t+1}) = 0 | \chi_{B_\varepsilon}(\Phi_t) = 0\}\\
&\qquad + p\{\chi_{B_\varepsilon}(\Phi_{t+1}) = 0 | \chi_{B_\varepsilon}(\Phi_t) \neq 0\})\\
&\quad = \prod_{t=1}^{k_{j+1}-1} p\{\chi_{B_\varepsilon}(\Phi_{t+1}) = 0 | \chi_{B_\varepsilon}(\Phi_t) = 0\}.\\
&\quad \le \prod_{t=1}^{j} p\{\chi_{B_\varepsilon}(\Phi_{k_{t+1}}) = 0 | \chi_{B_\varepsilon}(\Phi_{k_{t+1}-1}) = 0\}.
\end{aligned} \tag{36}$$

Because the time homogenous property of Markov Chain assures that

$$\begin{aligned}
&p\{\chi_{B_\varepsilon}(\Phi_{k_{j+1}}) = 0 | \chi_{B_\varepsilon}(\Phi_{k_{j+1}-1}) = 0\}\\
&\quad = p\{\chi_{B_\varepsilon}(\Phi_{k_2}) = 0 | \chi_{B_\varepsilon}(\Phi_{k_2-1}) = 0\}, \forall j.
\end{aligned} \tag{37}$$

In order to calculate $p\{\chi_{B_\varepsilon}(\Phi_{k_2}) = 0 | \chi_{B_\varepsilon}(\Phi_{k_2-1}) = 0\}$, we define

$$\Delta = \{\Phi | \chi_{B_\varepsilon}(\Phi) = 0\} \subset R^{4\cdot D\cdot Np}.$$

$\forall \Phi \in \Delta$, From Lemma 5, we can get

$$\begin{aligned}
&p\{\chi_{B_\varepsilon}(\Phi_{k_2})\\
&\quad = 0 | \chi_{B_\varepsilon}(\Phi_{k_2-1}) = 0\} = p\{\chi_{B_\varepsilon}(\Phi_{k_2}) = 0 | \Phi_{k_2-1} = \Phi\}\\
&\quad = 1 - p\{\chi_{B_\varepsilon}(\Phi_{k_2}) \neq 0 | \Phi_{k_2-1} = \Phi\}\\
&\quad \le 1 - \rho
\end{aligned} \tag{38}$$

From Eqs. (36) and (37), we know that the probability (35) becomes

$$p\{\chi_{B_\varepsilon}(\Phi_{k_{j+1}}) = 0\} \le (1-\rho)^j \quad .$$

This implies that

$$\begin{aligned}
&p\{\chi_{B_\varepsilon}(\Phi_{k_{j+1}}) \neq 0\} = 1 - p\{\chi_{B_\varepsilon}(\Phi_{k_{j+1}}) = 0\}\\
&\quad \ge 1 - (1-\rho)^j
\end{aligned}$$

By Lemma 3, one has

$$p\{p_g^{k_{j+1}} \in B_\varepsilon\} = p\left\{\chi_{B_\varepsilon}(\Phi_{k_{j+1}}) \neq 0\right\} \ge 1 - (1-\rho)^j.$$

Therefore, we have

$$\lim_{j\to\infty} p\left\{p_g^{k_{j+1}} \in B_\varepsilon\right\} \ge 1 - \lim_{j\to\infty} (1-\rho)^j = 1.$$

Thus

$$\lim_{j\to\infty} p\left\{p_g^{k_{j+1}} \in B_\varepsilon\right\} = 1.$$

By standard PSO algorithm, we know that $\{f(P_g^k)\}$ is monotone-decreasing sequence. Hence, one has

$$\lim_{k\to\infty} p\{p_g^k \in B_\varepsilon\} = 1.$$

This completes the proof of the theorem. □

Theorem 3 *Suppose that $\forall(\Phi_1, \Phi_2, \Phi_3, \ldots) \in \Omega$ is a sequence generated by standard PSO algorithm. If $x^* = (x_1^*, x_2^*, \ldots x_D^*)$ does not belong to $cl\left(\bigcup_{k=1}^{\infty}\bigcup_{i=1}^{Np} M_i^k\right)$ which is closure of the set $\bigcup_{k=1}^{\infty}\bigcup_{i=1}^{Np} M_i^k$, then there exists $\varepsilon > 0$ such that*

$$p\{p_g^k \in B_\varepsilon\} = 0 \quad \textit{for} \quad \forall k \ge 1. \tag{39}$$

Proof Since $cl\left(\bigcup_{k=1}^{\infty}\bigcup_{i=1}^{Np} M_i^k\right)$ is closed set, the objective function f, based on its continuity, has the minimum $f_{\min}$ on the set $cl\left(\bigcup_{k=1}^{\infty}\bigcup_{i=1}^{Np} M_i^k\right)$. By $x^* = (x_1^*, x_2^*, \ldots x_D^*) \notin cl\left(\bigcup_{k=1}^{\infty}\bigcup_{i=1}^{Np} M_i^k\right)$, we know that $f_{\min} - f(x^*) > 0$. Let $\varepsilon = f_{\min} - f(x^*)$, then $B_\varepsilon \bigcap cl\left(\bigcup_{k=1}^{\infty}\bigcup_{i=1}^{Np} M_i^k\right)$ is an empty set. This implies that $B_\varepsilon \bigcap M_i^k$ is also an empty set for $\forall i \in \{1, 2, \ldots, Np\}$ and $\forall k \ge 1$.

Because the random variables $x_{i1}^k, x_{i2}^k, \ldots, x_{iD}^k$ are independent, the joint density function of the D-dimensional random variable $x_i^k = (x_{i1}^k, x_{i2}^k, \ldots, x_{iD}^k)^{\mathrm{T}}$ is

$$\begin{aligned}
&p_{x_i^k}(u_1, u_2, \ldots, u_D) = p_{x_{i1}^k}(u_1) \cdot p_{x_{i2}^k}(u_2) \cdot \ldots \cdot p_{x_{iD}^k}(u_D)\\
&\quad \text{for} \quad \forall i \in \{1, 2, \ldots, Np\} \text{ and } k \ge 1.
\end{aligned} \tag{40}$$

If $\forall(u_1, u_2, \ldots, u_D) \notin M_i^k$, then there exists at least $d \in \{1, 2, \ldots, D\}$ such that $u_d \notin [\alpha_{id}^k, \tau_{id}^k]$. We have $p_{x_{id}^k}(u_d) = 0$ by Eqs. (6), (19), (20) and (21). From Eq. (40), we know that if $(u_1, u_2, \ldots, u_D)^{\mathrm{T}} \notin M_i^k$ then $p_{x_i^k}(u_1, u_2, \ldots, u_D) = 0$ for $\forall i \in \{1, 2, \ldots, Np\}$ and $k \ge 1$.

Since $B_\varepsilon \bigcap M_i^k$ is an empty set, one has

$$p\{x_i^k \in B_\varepsilon\} = \int_{B_\varepsilon} p_{x_i^k}(u_1, u_2, \ldots, u_D)\mathrm{d}u_1\mathrm{d}u_2 \ldots \mathrm{d}u_D = 0 \quad \text{for} \quad \forall i \in \{1, 2, \ldots, Np\} \text{ and } k \geq 1. \tag{41}$$

By standard PSO algorithm, we know that p_g^k is an element of the set $\bigcup_{t=1}^{k} \bigcup_{i=1}^{Np} \{x_i^t\}$ for $\forall k \geq 1$. Therefore, we have from Eq. (41) that

$$p\{p_g^k \in B_\varepsilon\} = 0 \quad \text{for} \quad \forall k \geq 1.$$

The proof is completed. □

We know from Theorem 2 that the standard PSO algorithm converges to the ε-optimum solution with probability one for any ε under certain conditions. But Theorem 3 shows that the standard PSO algorithm reaches the ε-optimum solution with probability zero for some ε. In order to overcome this drawback, we give a new improved PSO algorithm which converges to the ε-optimum solution with probability one in next section.

5 Improved PSO algorithm and Numerical simulations

5.1 Improved PSO algorithm

It is generally known that the effective methods of enhancing the performance of PSO algorithm are to balance the exploration and exploitation abilities. The exploitation ability is implemented for the best particles (the best fitness value) and exploration ability for the worst particles (the worst fitness value). Moreover, Gaussian perturbation is introduced in order to ensure that the proposed PSO algorithm converges to the ε-optimum solution with probability one for any ε. Based on this idea, we propose an improved PSO algorithm (Here, we call it IPSO) according to the following strategy.

At iteration k, the ith particle's best position $P_i^k = (p_{i1}^k, p_{i2}^k, \ldots, p_{iD}^k)$ is updated as follows:

$$\bar{p}_{id}^k = \begin{cases} \mu p_{id}^k (u_d - l_d)\rho_1 N(0, \sigma^2) \\ \quad + (1-\mu)(p_{id}^k + \rho_2 N(0, \sigma^2)), & \text{if rand} < \mathrm{CR} \\ p_{id}^k, & \text{otherwise} \end{cases}. \tag{42}$$

The first part of Eq. (42) is exploration operator which can improve exploration ability; the Second part of Eq. (42) is exploitation operator which can improve exploitation ability. Here, ρ_1 and ρ_2 are two positive real numbers, *rand* is a random number between 0 and 1. μ is the weighting factor, which is defined as follows:

$$\mu = \frac{f(P_i^k) - f\left(P_g^k\right)}{f_{\max} - f\left(P_g^k\right)}, \tag{43}$$

where $f_{\max} = \max_{1 \leq i \leq Np} \{f(P_i^k)\}$. $N(0, \sigma^2)$ represents Gaussian distribution with a time-varying standard deviation as follows:

$$\sigma = \sigma_{\max} - (\sigma_{\max} - \sigma_{\min}) \frac{k}{K_{\max}}, \tag{44}$$

where $\sigma_{\max}$ and $\sigma_{\min}$ are the initial and final values of standard deviation, k is the current iteration number, $K_{\max}$ is the maximum iteration number, and CR is a threshold which will change according to the following equation

$$\mathrm{CR} = \begin{cases} \frac{\alpha(f_{\mathrm{ave}} - f(P_i^k))}{f_{\mathrm{ave}} - f\left(P_g^k\right)}, & \text{if} \quad f(P_i^k) \leq f_{ave} \\ \frac{\alpha(f(P_i^k) - f_{\mathrm{ave}})}{f_{\max} - f_{ave}}, & \text{if} \quad f(P_i^k) > f_{ave} \end{cases}, \tag{45}$$

where α is a constant between 0 and 1, and f_{ave} is defined as follows:

$$f_{\mathrm{ave}} = \frac{1}{Np} \sum_{j=1}^{Np} f(P_j^k), \tag{46}$$

By Eqs. (42) and (45), we know that the best particle and the worst particle can be updated with a greater probability, the best particle can exploit around itself, and the worst particle is able to explore the search space thoroughly. If $\bar{p}_{id}^k$ is less than l_d or greater than u_d, then $\bar{p}_{id}^k = p_{id}^k$. The new best position of particle i at iteration k is updated as follows:

$$P_i^k(new) = \begin{cases} \bar{P}_i^k, & \text{if} \quad f(\bar{P}_i^k) < f(P_i^k) \\ P_i^k, & \text{if} \quad f(\bar{P}_i^k) \geq f(P_i^k) \end{cases}. \tag{47}$$

which will replace P_i^k. The pseudo-code of the IPSO algorithm is shown in Fig. 1.

5.2 Convergence analysis of IPSO

Lemma 6 *If $\bar{P}_i^k \in B_\varepsilon$, then $P_i^k(new) \in B_\varepsilon$ for any i and k.*

Proof If $P_i^k(new) = P_i^k$, then we know that $f(\bar{P}_i^k) \geq f(P_i^k)$ by Eq. (47). Since $\bar{P}_i^k \in B_\varepsilon$, we have $P_i^k \in B_\varepsilon$. Thus, we get $P_i^k(new) \in B_\varepsilon$. If $P_i^k(new) = \bar{P}_i^k$, then $P_i^k(new) \in B_\varepsilon$ is obvious. □

Lemma 7 *$\forall \Phi \in \Phi_{Np}$, if $\Phi_k = \Phi$, then there exists a constant $\rho > 0$, such that*

$$p(x_{B_\varepsilon}\left(\Phi_{k+1}\right) \neq 0) \geq \rho. \tag{48}$$

where Φ_{k+1} is generated by the IPSO algorithm.

standard deviation as follows:

```
Begin IPSO algorithm
    Initialize position $X_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$ and velocity $V_i = (v_{i1}, v_{i2}, \cdots, v_{iD})$, $i = 1,2,\cdots,Np$;
    Evaluate $f(X_i)$;
    Let particle's previous best $P_i = X_i$ and its fitness value $FP_i = f(X_i)$, $i = 1,2,\cdots,Np$;
    Get the global best $P_g$ and its fitness value $FP_g$.
    For k=1 to $K_{\max}$
        For $i = 1$ to $N_P$
            For $d = 1$ to $D$
                Calculate $\omega$ using Eq. (3);
                Update velocities using Eq. (1);
                Validate for velocity boundaries based on $V$;
                Update positions using Eq. (2);
                Validate for position boundaries based on $S$;
            End For
            If $f(X_i) < FP_i$ then
                $P_i = X_i$;
                $FP_i = f(X_i)$;
            End
            If $f(X_i) < FP_g$ then
                $P_g = X_i$;
                $FP_g = f(X_i)$;
            End If
        End For
        Calculate $f_{ave}$ and $\sigma$ according to Eqs (46) and (44);
        For $i = 1$ to $N_P$
            Calculate $CR$ and $\mu$ according to Eqs (45) and (43);
            For $d = 1$ to $D$
                Calculate $\overline{P}_{id}$ using Eq. (42);
            End For
            If $f(\overline{P}_i) < f(P_i)$ then
                $P_i = \overline{P}_i$;
                $FP_i = f(\overline{P}_i)$;
            End If
        End For
    End For
```

Fig. 1 Pseudo-code for the IPSO algorithm

Proof By IPSO algorithm, Eq. (22) and Lemma 6, we have

$$\begin{aligned} p(x_{B_\varepsilon}\left(\Phi_{k+1}\right) \neq 0) &\geq p(P_g^{k+1}(new) \in B_\varepsilon) \\ &\geq p(\bar{P}_g^k \in B_\varepsilon) = p((\bar{p}_{g1}^k, \bar{p}_{g2}^k, \ldots, \bar{p}_{gD}^k) \in B_\varepsilon) \end{aligned}$$

From Eqs. (43) and (45), we know that $\mu = 0$ and CR $= \alpha$ when $p_{id}^k = p_{gd}^k$. In other words, $\bar{p}_{gd}^k = p_{gd}^k + \rho_2 N(0, \sigma^2), d = 1, 2, \ldots, D$ is implemented with probabilityα. Thus, $\bar{p}_{gd}^k$ is a random variable, and its probability density function is

$$p_{\bar{p}_{gd}^k}(x_d) = \frac{1}{\sqrt{2\pi}\sigma\rho_2} e^{-\frac{(x_d - p_{gd}^k)^2}{2\sigma^2\rho_2^2}}, \text{ for } x_d \in (-\infty, +\infty).$$

Since $\bar{p}_{g1}^k, \bar{p}_{g2}^k, \ldots, \bar{p}_{gD}^k$ are mutually independent random variables, the probability density function of the D-random variable $(\bar{p}_{g1}^k, \bar{p}_{g2}^k, \ldots, \bar{p}_{gD}^k)$ is

$$\begin{aligned} p(x_1, x_2, \ldots, x_D) &= \prod_{d=1}^{D} \frac{1}{\sqrt{2\pi}\sigma\rho_2} e^{-\frac{(x_d - p_{gd}^k)^2}{2\sigma^2\rho_2^2}} \\ &= \left(\frac{1}{\sqrt{2\pi}\sigma\rho_2}\right)^D e^{-\sum_{d=1}^{D} \frac{\left(x_d - p_{gd}^k\right)^2}{2\sigma^2\rho_2^2}}, \text{ for } x_d \in (-\infty, +\infty). \end{aligned}$$

Thus,

$$\begin{aligned} & p((\bar{p}_{g1}^k, \bar{p}_{g2}^k, \ldots, \bar{p}_{gD}^k) \in B_\varepsilon) \\ &= \alpha^D \int_{B_\varepsilon} \left(\frac{1}{\sqrt{2\pi}\sigma\rho_2}\right)^D e^{-\sum\limits_{d=1}^{D} \frac{(x_d - p_{gd}^k)^2}{2\sigma^2\rho_2^2}} \mathrm{d}x_1 \mathrm{d}x_2 \ldots \mathrm{d}x_D \\ &\geq \left(\frac{\alpha}{\sqrt{2\pi}\sigma_{\max}\rho_2}\right)^D e^{-D\frac{2m^2}{\sigma_{\min}^2\rho_2^2}} \mu(B_\varepsilon). \end{aligned}$$

Let $\rho = \left(\frac{\alpha}{\sqrt{2\pi}\sigma_{\max}\rho_2}\right)^D e^{-D\frac{2m^2}{\sigma_{\min}^2\rho_2^2}} \mu(B_\varepsilon) > 0$, we have

$$p(x_{B_\varepsilon}\left(\Phi_{k+1}\right) \neq 0) = p\left(x_{B_\varepsilon}\left(\Phi_{k+1}\right) \neq 0 | \Phi_k {=} \Phi\right) \geq \rho {>} 0.$$

This completes the proof of lemma. □

Theorem 4 *Suppose that* $\forall(\Phi_1, \Phi_2, \Phi_3, \ldots) \in \Omega$ *is a sequence generated by IPSO algorithm. Then the sequence* $\{p_g^k\}$ *converges to the* ε*-optimum solution with probability one, i.e.,*

$$\lim_{k \to \infty} p\{p_g^k \in B_\varepsilon\} = 1.$$

Proof See the proof of Theorem 2. □

Table 1 Unimodal test functions

Test function	S	$f_{\text{opt.}}$
$F_1(x) = \sum\limits_{i=1}^{D} x_i^2$	$[-100, 100]^n$	0
$F_2(x) = \sum\limits_{i=1}^{D} \|x_i\| + \prod\limits_{i=1}^{D} \|x_i\|$	$[-10, 10]^D$	0
$F_3(x) = \sum\limits_{i=1}^{D} \left(\sum_{j=1}^{i} x_j\right)^2$	$[-100, 100]^D$	0
$F_4(x) = \max_i \{\|x_i\|, 1 \leq i \leq D\}$	$[-100, 100]^D$	0
$F_5(x) = \sum\limits_{i=1}^{D-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$	$[-30, 30]^D$	0
$F_6(x) = \sum\limits_{i=1}^{D} ([x_i + 0.5])^2$	$[-100, 100]^D$	0
$F_7(x) = \sum\limits_{i=1}^{D} i x_i^4 + random\,[0, 1)$	$[-1.28, 1.28]^D$	0

5.3 Numerical simulations

5.3.1 Benchmark functions

In order to evaluate the performance of the proposed IPSO algorithm, we take the 13 standard benchmark functions provided in Yao et al. (1999). These functions are given in Tables 1 and 2, respectively, where D represents the dimension of the function, S is the search region, and f_{opt} is the minimum value of the function. The functions in Table 1 are unimodal and in Table 2 are multimodal. The multimodal functions have a large number of local minima, so the algorithm must be the capable of finding the optimum solution and should not be trapped in local optima.

5.3.2 Comparison with other algorithms

The IPSO algorithm is applied to 13 standard benchmark functions, and the results is compared with LDIWPSO (Shi and Eberhart 1999), AIWPSO (Nickabadi et al. 2011), CDIWPSO (Feng et al. 2007), DAPSO (Shen et al. 2010), and SSRDIWPSO (Adewumi and Arasomwan 2016). In our experiments, the same set of parameters is assigned for all algorithms: The initial and final values of inertia weight are 0.9 and 0.4, the acceleration factors are $c_1 = 2$ and $c_2 = 2$, the search region of velocities of the particles is set to S which is the search region of positions of the particles, population size is set to 30, and the maximum number of iteration $K_{\max}$ is 3000. In IPSO, other parameters are set as follows: $\rho_1 = 20$, $\rho_2 = 0.5$, $\sigma_{\max} = 1$, $\sigma_{\min} = 0.2$, and $\alpha = 0.5$. In all algorithms, if the positions and velocities of the particles are beyond their search region, then the positions and velocities of the particles are updated by using random method. For all the algorithms, two stopping conditions are adopted, the one is that the maximum number of iteration is satisfied and the other one is that $|f(p_g) - f_{\text{opt}}| < \varepsilon$ or the iteration numbers are over the maximum number of iteration, where ε is a small tolerance. The first stopping condition is used for evaluating the solution accuracy, and the second is used to evaluate the runtimes of the algorithms. All algorithms are implemented in MATLAB 7.0 and run on PC with 2.00 GB RAM memory, 2.10 GHz CPU, and Windows 7 operation system. For each problem, the experiment is repeated 30 times.

By the first stopping condition, Tables 3, 4, 5, 6, 7, and 8 show the results with respect to the best of the best solutions, worst of the best solutions, average of the best solutions, standard deviation of the best solutions, and average runtimes of algorithms for the unimodal and multimodal functions with dimensions 30, 100, and 1000, respectively. In Table 7, “f” means that the running results fail.

In Table 3, we see that the six algorithms can find better optimum solutions on functions F_1 and F_2, but AIWPSO and SSRDIWPSO outperform IPSO. For functions F_3 and F_4, LDIWPSO, AIWPSO, CDIWPSO, DAPSO, and SSRDIWPSO have a very poor ability to explore and exploit the search space, whereas IPSO can get better optimum solutions. This is clear indication that IPSO has better global search ability and can easily get out of local optima than the other five algorithms. For function F_5, the six algorithms does not find the optimum solutions, but the average value of the best solutions obtained by IPSO is better than other algorithms, and IPSO is more robust than the other algorithms based on comparing the obtained standard deviation. For function F_6, the six algorithms get same results. For function F_7, IPSO is superior to the other five algorithms.

Table 4 shows that IPSO has a strong ability to explore and exploit the search space except for the function F_{13}. For functions F_8, F_9, and F_{11}, IPSO is much better than the other five algorithms in exploring and exploiting and can find better optimum solutions. For function F_{10}, IPSO is worse than CDIWPSO, but can also get better optimum solutions and is superior to the other four algorithms. For function F_{12}, IPSO is worse than AIWPSO, CDIWPSO, and SSRDIWPSO for the best of the best solutions, but it is much better than the other five algorithms for the worst and average of the best solutions and more robust than the other algorithms based on comparing the obtained standard deviation. For function F_{13}, AIWPSO, CDIWPSO, and SSRDIWPSO are superior to IPSO for the best of the best solutions, but all algorithms act nearly the same for the average of the best solutions.

Table 5 shows the strong ability of our method to search for optimum solutions. Except for function F_5, IPSO performs significantly better than the other five algorithms in exploring and exploiting the search space, and it exactly leads to finding the optimum solutions for function F_6. For function F_5, IPSO do not find the optimum solutions, but IPSO is better than the other five algorithms.

Table 2 Multimodal test functions

Test function	S	$f_{\text{opt.}}$
$F_8(x) = \sum_{i=1}^{D} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	$[-500, 500]^D$	$-418.9829 \times D$
$F_9(x) = \sum_{i=1}^{D-1} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$[-5.12, 5.12]^D$	0
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)$ $+ 20 + e$	$[-32, 32]^D$	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^D$	0
$F_{12}(x) = \frac{\pi}{D}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2\right.$ $\left.\left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_D - 1)^2\right\}$ $+ \sum_{i=1}^{D} u(x_i, 10, 100, 4)$	$[-50, 50]^D$	0
$y_i = 1 + \frac{x_i+1}{4}, u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$		
$F_{13}(x) = 0.1\left\{\text{in}^2(3\pi x_1) + \sum_{i=1}^{D}(x_i - 1)^2\left[1 + \sin^2(3\pi x_i + 1)\right]\right.$ $\left.+(x_D - 1)^2\left[1 + \sin^2(2\pi x_D)\right]\right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	$[-50, 50]^D$	0

Table 3 Experimental results of benchmark functions in Table 1 with $D = 30$ and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_1	Best best-so-far	3.7200e−018	2.4378e−059	8.1387e−039	5.6880e−022	1.4644e−050	1.0166e−024
	Worst best-so-far	1.0396e−015	7.8901e−053	6.1080e−026	1.0645e−013	4.4389e−041	2.2898e−021
	Average best-so-far	2.1221e−016	8.3201e−054	3.2822e−027	6.4135e−015	2.2208e−042	2.3703e−022
	Std best-so-far	9.8689e−032	3.9513e−106	1.8600e−052	5.7203e−028	9.8511e−083	3.7844e−043
	Average time	1.8627	1.8339	1.9048	1.9482	1.9255	2.6294
F_2	Best best-so-far	1.3203e−012	7.1286e−034	1.1676e−018	3.8006e−014	1.2939e−025	7.5160e−015
	Worst best-so-far	1.8845e−008	3.3910e−026	4.3698e−011	9.9588e−010	4.1432e−019	1.2107e−013
	Average best-so-far	9.5626e−010	1.9941e−027	2.4152e−012	9.9521e−011	2.5891e−020	405899e−014
	Std best-so-far	1.7729e−017	5.7166e−053	9.5216e−023	7.1478e−020	8.6290e−039	9.9829e−028
	Average time	1.9837	1.9083	1.9186	1.9666	1.9705	2.7004
F_3	Best best-so-far	38.4390	19.5181	0.3708	20.1182	0.1488	1.6134e−007
	Worst best-so-far	127.1990	857.6853	23.2221	186.6040	1.7661	4.2050e−006
	Average best-so-far	68.8021	324.0082	5.0969	74.8657	0.8713	9.6916e−007
	Std best-so-far	632.1191	7.5609e+004	22.7318	2.4965e+003	0.2057	9.1200e−013
	Average time	5.6536	5.4785	5.6104	5.7670	5.7857	8.2669
F_4	Best best-so-far	0.6536	1.5688	0.2949	1.1092	0.1122	1.8967e−004
	Worst best-so-far	4.0583	6.2668	1.9690	5.1332	0.9092	7.0414e−004
	Average best-so-far	2.1339	3.7650	0.9590	2.2817	0.4309	4.3207e−004
	Std best-so-far	0.7476	1.9191	0.2045	0.9574	0.0615	2.0419e−008
	Average time	1.9269	1.8478	1.8970	1.9142	1.8909	2.6544
F_5	Best best-so-far	1.4469	1.1210	6.2043	13.1903	0.1676	23.8094
	Worst best-so-far	96.4663	223.6793	81.0026	235.1995	81.1743	24.2310
	Average best-so-far	43.2518	47.2351	34.7325	57.8589	25.2189	24.0819
	Std best-so-far	1.0098e+003	2.5389e+003	716.4503	3.6466e+003	570.8850	0.0154
	Average time	2.1457	2.1561	2.1275	2.1786	2.1540	3.2922
F_6	Best best-so-far	0	0	0	0	0	0
	Worst best-so-far	0	0	0	0	0	0
	Average best-so-far	0	0	0	0	0	0
	Std best-so-far	0	0	0	0	0	0
	Average time	1.9655	1.7832	1.9780	1.9693	1.9449	2.5439
F_7	Best best-so-far	0.0102	0.0206	0.0068	0.0057	0.0072	1.1685e−005
	Worst best-so-far	0.0330	0.0902	0.0190	0.0240	0.0149	4.2448e−004
	Average best-so-far	0.0181	0.0415	0.0124	0.0146	0.0109	1.6423e−004
	Std best-so-far	3.9040e−005	3.7490e−004	1.8898e−005	2.8161e−005	7.2492e−006	1.8402e−008
	Average time	2.9729	2.7924	2.9138	2.9228	2.8661	4.1544

Table 4 Experimental results of benchmark functions in Table 2 with $D = 30$ and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_8	Best best-so-far	−9.6677e+003	−1.0353e+004	−9.7072e+003	−9.6874e+003	−9.0558e+003	−1.2326e+004
	Worst best-so-far	−8.6018e+003	−7.9557e+003	−8.2267e+003	−7.7924e+003	−7.2593e+003	−1.0411e+004
	Average best-so-far	−9.0745e+003	−9.2402e+003	−8.8988e+003	−8.7616e+003	−8.1645e+003	−1.1505e+004
	Std best-so-far	8.1482e+004	4.8783e+005	1.2541e+005	2.3959e+005	2.3108e+005	1.4787e+005
	Average time	2.5695	2.4796	2.5883	2.6316	2.1644	3.6874
F_9	Best best-so-far	21.8891	16.9143	19.8992	21.8891	19.8992	0
	Worst best-so-far	47.7580	69.6470	52.7327	56.7126	50.7428	5.6843e−014
	Mean best-so-far	34.7738	46.7630	31.9879	33.7291	34.8235	1.1369e−014
	Std best-so-far	66.5319	163.7045	95.5839	73.7661	77.1109	5.4420e−028
	Average time	2.1708	2.0441	2.1309	2.2143	2.1027	2.7264
F_{10}	Best best-so-far	2.6525e−010	2.2204e−014	1.5099e−014	1.6596e−011	7.9936e−015	5.0626e−014
	Worst best-so-far	4.3373e−008	2.8138	2.9310e−014	1.5017	0.9313	6.0014e−012
	Average best-so-far	6.2393e−009	1.0267	1.8474e−014	0.0751	0.0466	1.6380e−012
	Std best-so-far	1.4389e−016	0.9546	2.3217e−029	0.1128	0.0434	2.0417e−024
	Average time	2.2280	2.0708	2.1462	2.2574	2.0977	2.9551
F_{11}	Best best-so-far	0	0	0	0	0	0
	Worst best-so-far	0.0394	0.1177	0.0539	0.0712	0.0634	0
	Average best-so-far	0.0129	0.0373	0.0107	0.0187	0.0161	0
	Std best-so-far	1.2541e−004	0.0018	2.5114e−004	2.5829e−004	2.6579e−004	0
	Average time	2.1699	2.5014	2.6409	2.6898	2.1730	3.4836
F_{12}	Best best-so-far	5.467e−020	1.6109e−032	1.8045e−032	3.7042e−022	1.9900e−32	1.8120e−022
	Worst best-so-far	0.6219	1.6624	0.1037	0.2073	0.6227	7.8642e−019
	Average best-so-far	0.0777	0.1505	0.0104	0.0311	0.0933	7.1892e−020
	Std best-so-far	0.0259	0.1437	0.0010	0.0035	0.0248	3.1568e−038
	Average time	5.1608	5.0023	5.2787	5.2207	5.1299	6.5646
F_{13}	Best best-so-far	1.0716e−017	4.5545e−032	5.6639e−032	4.3786e−020	2.0893e−032	1.9036e−019
	Worst best-so-far	0.0110	0.0974	0.0110	0.0989	0.0548	0.0974
	Average best-so-far	0.0038	0.0076	0.0033	0.0082	0.0049	0.0060
	Std best-so-far	2.8910e−005	4.6973e−004	2.6686e−005	4.8127e−004	1.570e−004	4.7423e−004
	Average time	5.1324	5.1036	5.1663	5.2160	5.0834	7.1671

Table 5 Experimental results of benchmark functions in Table 1 with $D = 100$ and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_1	Best best-so-far	0.7866	0.0975	4.6212e−004	3.9320	6.9550e−006	4.7830e−008
	Worst best-so-far	8.1462	1.9808e+003	0.0437	65.3566	0.0183	2.9925e−007
	Average best-so-far	4.9813	267.7724	0.0144	13.7696	0.036	1.6206e−007
	Std best-so-far	6.2629	3.9162e+005	2.0829e−004	341.6476	3.8155e−005	4.4403e−015
	Average time	2.9881	3.0222	3.0502	3.0196	2.9612	5.6765
F_2	Best best-so-far	0.2999	0.0011	0.0117	0.3967	0.0066	5.2200e−004
	Worst best-so-far	1.0612	6.9421	0.7983	1.6537	0.4078	0.0014
	Average best-so-far	0.5532	0.8110	0.2202	0.7298	0.0847	8.8441e−004
	Std best-so-far	0.0620	4.6740	0.0660	0.1668	0.0162	8.8674e−008
	Average time	3.1006	3.1210	3.1493	3.1126	3.1054	5.5813
F_3	Best best-so-far	3.2344e+004	6.0036e+004	2.6844e+004	3.2888e+004	1.8457e+004	7.7686e−005
	Worst best-so-far	6.8893e+004	9.5209e+004	4.1553e+004	4.2624e+004	3.7090e+004	3.8951e−004
	Average best-so-far	4.2926e+004	7.2446e+004	3.3921e+004	3.8367e+004	2.6599e+004	2.2822e−004
	Std best-so-far	1.0373e+008	9.7502e+007	1.3685e+007	1.1646e+007	3.5236e+007	1.2299e−008
	Average time	17.7789	17.7395	17.8190	17.6647	17.5746	25.8889
F_4	Best best-so-far	31.9994	30.5903	27.4564	26.5518	25.5555	0.0020
	Worst best-so-far	37.6798	42.1095	33.4072	33.4156	35.0141	0.0034
	Average best-so-far	35.1506	38.0418	31.2513	30.3681	30.2952	0.0024
	Std best-so-far	2.8770	15.7458	3.2007	4.0288	9.3255	1.7143e−007
	Average time	3.0685	3.1325	3.0876	3.0933	3.0051	5.5482
F_5	Best best-so-far	1.0635e+003	409.3716	322.6039	1.1419e+003	178.7179	95.5883
	Worst best-so-far	2.2784e+003	6.5159e+005	490.0822	4.5432e+003	433.0322	96.0272
	Average best-so-far	1.7148e+003	1.0068e+005	416.7328	2.1963e+003	287.9769	95.7224
	Std best-so-far	1.2294e+005	4.7342e+010	3.0073e+003	1.1596e+006	4.3593e+003	0.0157
	Average time	3.3105	3.3445	3.4173	3.3616	3.3089	6.0300
F_6	Best best-so-far	29	240	3	34	6	0
	Worst best-so-far	58	2463	163	149	56	0
	Average best-so-far	39.1	922.6000	34.2000	61	22.9000	0
	Std best-so-far	67.6556	4.9445e+005	2.3964e+003	1.1644e+003	249.2111	0
	Average time	3.2119	3.1977	3.2292	3.2092	3.1267	4.6642
F_7	Best best-so-far	0.4670	6.9153	0.2796	0.35309	0.2556	6.0845e−006
	Worst best-so-far	0.7015	2.0303	0.9535	0.9074	0.6380	2.5307e−004
	Average best-so-far	0.5872	1.4511	0.4338	0.5875	0.3626	2.5307e−004
	Std best-so-far	0.0069	0.1321	0.0380	0.0349	0.0133	7.2054e−008
	Average time	6.1237	6.1778	6.1794	6.2018	6.1033	9.4619

Table 6 Experimental results of benchmark functions in Table 2 with $D = 100$ and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_8	Best best-so-far	−2.4203e+004	−2.4325e+004	−2.6023e+004	−2.3279e+004	−2.4290e+004	−2.7790e+004
	Worst best-so-far	−2.1688e+004	−1.9073e+004	−2.2138e+004	−2.0573e+004	−2.0953e+004	−2.3531e+004
	Average best-so-far	−2.3447e+004	−2.2782e+004	−2.4006e+004	−2.2300e+004	−2.2912e+004	−2.5947e+004
	Std best-so-far	5.9049e+005	3.1082e+006	9.7368e+005	8.5823e+005	1.2685e+006	1.9923e+006
	Average time	3.8869	5.3293	5.2779	5.2879	3.7918	8.3848
F_9	Best best-so-far	178.8102	229.0908	155.2209	173.8396	196.0066	**1.7372e−005**
	Worst best-so-far	236.9478	364.2130	254.7276	251.8596	246.7495	**7.1269e−004**
	Mean best-so-far	202.5107	288.3807	207.4560	205.2461	221.9752	**2.2287e−004**
	Std best-so-far	431.1648	1.7964e+003	600.2752	713.6248	357.3478	7.0131e−008
	Average time	3.7864	3.7005	3.7556	3.8122	3.6094	5.6833
F_{10}	Best best-so-far	1.8438	2.1139	0.0460	0.6755	1.2810	**4.7543e−005**
	Worst best-so-far	3.1610	6.7167	2.5295	2.4543	3.2402	**1.8373e−004**
	Average best-so-far	2.6361	3.6532	1.6410	1.7576	2.1490	**1.1072e−004**
	Std best-so-far	0.1386	1.4072	0.4193	0.2407	0.3557	**1.8095e−009**
	Average time	3.9794	3.8663	3.9377	3.9292	3.7305	5.3207
F_{11}	Best best-so-far	0.5535	0.1839	5.7285e−004	0.6584	1.2848e−005	**1.3479e−009**
	Worst best-so-far	1.4761	9.0210	0.8887	1.5441	0.2359	**1.5598e−008**
	Average best-so-far	0.9360	2.4371	0.1507	1.0311	0.0431	**5.4281e−009**
	Std best-so-far	0.0720	9.1813	0.0735	0.0674	**0.0053**	1.6022e−017
	Average time	4.0171	5.4150	5.2367	5.4304	3.7128	8.2233
F_{12}	Best best-so-far	4.8069	14.2203	0.8769	3.9389	0.2204	**8.3972e−005**
	Worst best-so-far	18.3680	3.6951e+003	7.7261	64.5938	1.9714	**4.6493e−004**
	Average best-so-far	10.0208	1.5335e+003	3.4659	13.0809	1.2037	**2.1407e−004**
	Std best-so-far	16.4510	1.7375e+006	3.6453	336.6386	0.4831	1.5031e−008
	Average time	12.0758	12.0172	11.9815	12.0748	11.9103	17.5980
F_{13}	Best best-so-far	125.8391	169.0794	41.6924	101.4791	7.4490	1.8613
	Worst best-so-far	905.5386	3.5832e+005	85.8243	254.2186	101.8327	3.3798
	Average best-so-far	335.1282	1.0427e+005	64.7075	162.0313	44.1807	2.7256
	Std best-so-far	5.8513e+004	1.7149e+010	222.7063	1.8399e+003	965.5963	0.2743
	Average time	12.0563	12.0771	11.9889	12.0665	11.9448	17.5691

Table 7 Experimental results of benchmark functions in Table 1 with $D = 1000$ and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_1	Best best-so-far	3.005e+005	2.6688e+005	1.7117e+005	2.4063e+005	1.5665e+005	**7.5897e−004**
	Worst best-so-far	3.3424e+005	4.2915e+005	2.2189+005	2.6287e+005	2.2043e+005	**0.0011**
	Average best-so-far	3.1849e+005	3.2811e+005	1.9442e+005	2.5113e+005	1.8804e+005	**9.3017e−004**
	Std best-so-far	1.0221e+008	2.4921e+009	2.7833e+008	6.8843e+007	2.8131e+008	1.2687e−008
	Average time	46.8835	48.1119	47.0162	46.8394	47.1211	118.0252
F_2	Best best-so-far	f	f	f	f	f	f
	Worst best-so-far	f	f	f	f	f	f
	Average best-so-far	f	f	f	f	f	f
	Std best-so-far	f	f	f	f	f	f
	Average time	48.0368	50.8487	48.8273	196.1318	48.5534	61.7963
F_3	Best best-so-far	4.4091e+006	6.4031e+006	4.5248e+006	4.1950e+006	3.0557e+006	**0.0230**
	Worst best-so-far	6.9739e+006	9.1063e+006	7.2624e+006	7.2630e+006	5.4328e+006	**0.0533**
	Average best-so-far	5.6511e+006	8.0246e+006	5.2072e+006	5.4483e+006	4.2604e+006	**0.0390**
	Std best-so-far	6.4352e+011	1.0791e+012	6.5525e+011	7.5099e+011	8.5913e+011	1.0617e−004
	Average time	971.5260	1219.5157	973.5021	971.5174	978.7052	1888.8560
F_4	Best best-so-far	79.5717	79.5244	79.0658	77.2428	77.9427	**0.0032**
	Worst best-so-far	84.3496	84.9844	82.3363	84.3209	82.8618	**0.0044**
	Average best-so-far	82.1660	83.2208	81.4240	80.8847	81.0078	**0.0038**
	Std best-so-far	2.2568	3.7017	1.2010	4.4449	3.2900	1.2841e−007
	Average time	47.3942	47.4161	47.4031	47.8274	47.2219	117.3926
F_5	Best best-so-far	4.3716e+008	3.2079e+008	2.0114e+008	1.7381e+008	2.0210e+008	998.4880
	Worst best-so-far	6.0613e+008	1.0991e+009	2.8191e+008	2.8833e+008	2.7985e+008	998.8497
	Average best-so-far	5.2612e+008	7.0663e+008	2.4885e+008	2.4728e+008	2.3164e+008	998.7345
	Std best-so-far	2.4784e+015	5.7321e+016	5.6417e+014	1.3960e+015	7.2085e+014	0.0142
	Average time	54.9438	53.9825	54.2090	53.8174	54.2444	133.7422
F_6	Best best-so-far	297225	298271	193324	238877	175188	**0**
	Worst best-so-far	347350	525815	223129	307097	216428	**0**
	Average best-so-far	3.2636e+005	4.0291e+005	2.0536e+005	2.6269e+005	1.9499e+005	**0**
	Std best-so-far	2.8900e+008	5.1694e+009	1.0989e+008	4.2711e+008	2.1865e+008	**0**
	Average time	48.9438	48.7067	49.2585	48.8090	48.9883	118.7252
F_7	Best best-so-far	6.5996e+003	6.8328e+003	2.6494e+003	2.1475e+003	2.3061e+003	**4.8408e−005**
	Worst best-so-far	7.7100e+003	1.4872e+004	3.9256e+003	3.5719e+003	3.4984e+003	**7.9337e−004**
	Average best-so-far	7.1038e+003	1.0422e+004	3.1474e+003	3.1527e+003	2.9690e+003	**3.1112e−004**
	Std best-so-far	1.7206e+005	9.7362e+006	1.2661e+005	2.0648e+005	1.3395e+005	5.6791e−008
	Average time	80.1818	80.1332	80.3776	79.9351	81.2054	161.3199

Table 8 Experimental results of benchmark functions in Table 2 with $D = 1000$ and $K_{\max} = 3000$

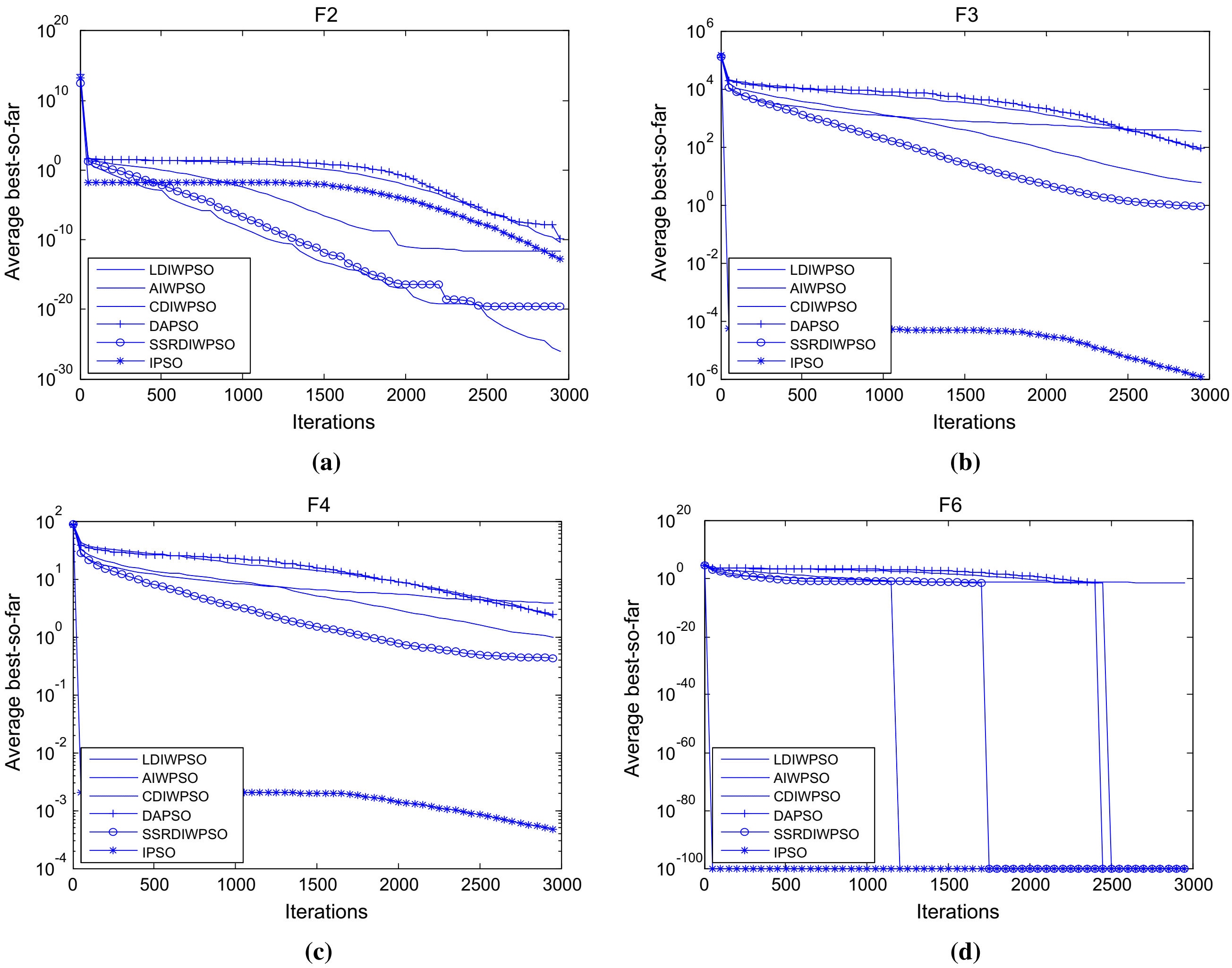
		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_8	Best best-so-far	−1.1797e+005	−1.1390e+005	−1.3865e+005	−1.2068e+005	−1.0978e+005	−1.4274e+005
	Worst best-so-far	−1.0400e+005	−8.3513e+004	−1.2945e+005	−1.0875e+005	−9.7131e+004	−1.2624e+005
	Average best-so-far	−1.1192e+005	−1.0249e+005	−1.3405e+005	−1.1384e+005	−1.0378e+005	−1.3405e+005
	Std best-so-far	1.6228e+007	1.0671e+008	8.3727e+006	1.7661e+007	1.9894e+007	2.6343e+007
	Average time	56.0586	69.5582	69.5078	69.4483	56.0285	136.9246
F_9	Best best-so-far	6.9158e+003	7.6403e+003	5.6489e+003	6.3234e+003	5.7810e+003	0.0693
	Worst best-so-far	7.4843e+003	9.8352e+003	6.4098e+003	7.1668e+003	6.1225e+003	0.1438
	Mean best-so-far	7.1712e+003	8.6997e+003	6.0835e+003	6.7021e+003	5.9629e+003	0.1126
	Std best-so-far	2.3502e+004	6.3900e+005	5.4764e+004	1.0159e+005	1.2870e+004	5.3447e-004
	Average time	54.7937	54.6627	54.7718	54.6620	54.7299	124.5108
F_{10}	Best best-so-far	16.4548	16.0983	16.0269	16.3948	15.8779	**0.0028**
	Worst best-so-far	17.2343	18.2017	17.0788	17.0999	17.2168	**0.0036**
	Average best-so-far	16.9062	17.3490	16.6070	16.7377	16.5031	**0.0033**
	Std best-so-far	0.0537	0.4779	0.1272	0.0571	0.1456	7.5896e-008
	Average time	57.7528	57.7654	57.9471	57.5939	57.5110	132.8086
F_{11}	Best best-so-far	2.6325e+003	2.6182e+003	1.5587e+003	2.0603e+003	1.4861e+003	**1.8921e−006**
	Worst best-so-far	3.1318e+003	4.7925e+003	2.0089e+003	2.6917e+003	1.8423e+003	**5.9980e−006**
	Average best-so-far	2.8482e+003	3.2720e+003	1.7954e+003	2.3173e+003	1.6685e+003	**4.0095e−006**
	Std best-so-far	2.2961e+004	3.6056e+005	1.8463e+004	4.8054e+004	8.5404e+003	1.7541e-012
	Average time	60.9706	74.5597	74.2099	74.3724	60.5574	159.3601
F_{12}	Best best-so-far	4.7997e+008	4.4647e+008	1.8841e+008	7.5148e+007	1.6763e+008	0.6404
	Worst best-so-far	8.0636e+008	1.7774e+009	3.2669e+008	2.0366e+008	2.9147e+008	0.9108
	Average best-so-far	6.6817e+008	9.5911e+008	2.6949e+008	1.3008e+008	2.1466e+008	0.7561
	Std best-so-far	1.0145e+016	2.5397e+017	1.7468e+015	1.4725e+015	1.8652e+015	0.0081
	Average time	148.4144	148.5472	149.0028	148.7707	149.0678	278.1809
F_{13}	Best best-so-far	1.5797e+009	1.1755e+009	5.6052e+008	3.6181e+008	4.0202e+008	98.4273
	Worst best-so-far	2.2188e+009	3.2730+009	8.4702e+008	7.5252e+008	8.3351e+008	99.8093
	Average best-so-far	1.8805e+009	2.1808e+009	6.9963e+008	5.1477e+008	6.1438e+008	99.0956
	Std best-so-far	4.2698e+016	3.5867e+017	1.1592e+016	1.6866e+016	1.7573e+016	0.1544
	Average time	149.1679	148.9309	149.3267	149.0598	149.3364	283.1745

As Table 6 illustrates, LDIWPSO, AIWPSO, CDIWPSO, DAPSO, and SSRDIWPSO have poor exploration and exploitation for the functions in Table 2, but IPSO has a good ability to explore and exploit the search space and can find good solutions on functions F_9, F_{10}, F_{11}, and F_{12}. For functions F_8 and F_{13}, IPSO also has poor performance in finding good solutions, but it is better than the other five methods.

Tables 7 and 8 show the comparative results on the high-dimensional benchmark functions of Tables 1 and 2. The dimension of these functions is 1000. From Tables 7 and 8, we can see that IPSO outperforms the other five algorithms for all functions. LDIWPSO, AIWPSO, CDIWPSO, DAPSO, and SSRDIWPSO have a very poor ability to explore and exploit the search space and cannot find good solutions for all functions, while IPSO has a better ability to find the good solutions on functions F_1, F_3, F_4, F_6, F_7, F_{10}, and F_{11}. Especially for function F_6, it exactly finds the global optimum solutions. IPSO is able to find acceptable solutions on functions F_9 and F_{12}. For functions F_5, F_8, and F_{13}, IPSO cannot find good solutions, but it is significantly better than other algorithms. For function F_2, all methods are not able to obtain the numerical results. It is noted that according to obtained the average best-so-far solutions, IPSO is much better than the other five methods in high-dimensional search problems.

From Tables 3–8, we can see that the average runtimes of LDIWPSO, AIWPSO, CDIWPSO, DAPSO, and SSRDIWPSO are nearly the same for all benchmark functions, while the average runtimes of IPSO are longer than those of other five algorithms for all benchmark functions. The mutation operator introduced in the standard particle swarm algorithm can promote the PSO to explore and exploit the search space, but the computational time will greatly increase at the same time.

Figure 2 shows the progress of the average best-so-far solution of LDIWPSO, AIWPSO, CDIWPSO, DAPSO, SSRDIWPSO, and IPSO over 30 runs for functions F_2, F_3, F_4, and F_6, with dimension 30. In Fig. 2, the convergence



(a) **(b)**

(c) **(d)**

Fig. 2 Performance comparison for functions **a** F_2, **b** F_3, **c** F_4, and **d** F_6 with $D = 30$ and $K_{\max} = 3000$

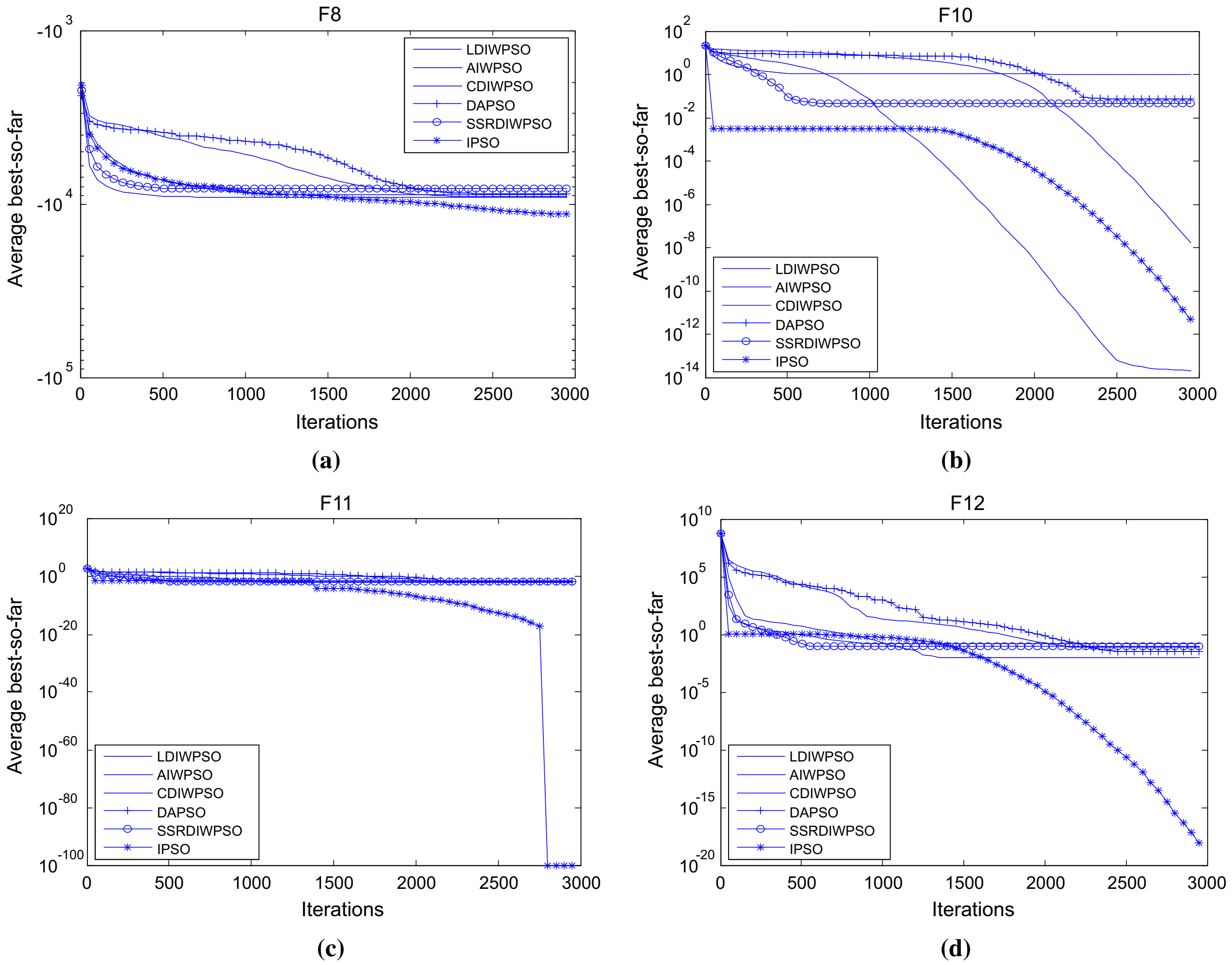


Fig. 3 Performance comparison for functions **a** F_8, **b** F_{10}, **c** F_{11} and **d** F_{12} with $D = 30$ and $K_{\max} = 3000$

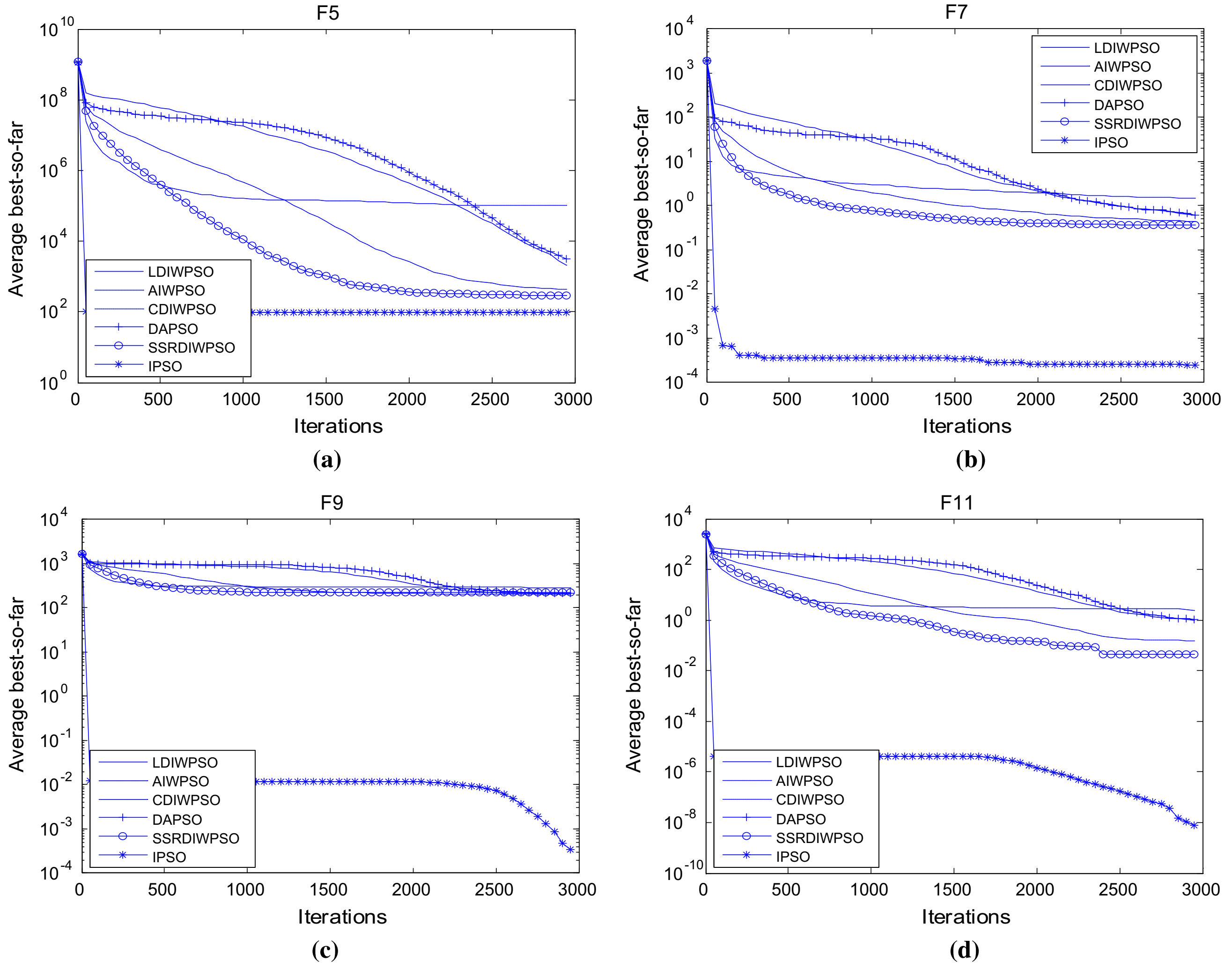
speed of IPSO is faster than that of the other five algorithms except for function F_2, and it achieves the better quality results for functions F_2, F_3, F_4, and F_6. In particular, the convergence speed of IPSO is very rapid for functions F_3, F_4, and F_6.

The progress of the average best-so-far solution of LDIWPSO, AIWPSO, CDIWPSO, DAPSO, SSRDIWPSO, and IPSO over 30 runs for functions F_8, F_{10}, F_{11}, and F_{12} with dimension 30 are shown in Fig. 3. From Fig. 3, compared against the other five algorithms, the convergence speed of IPSO is faster except for function F_{10}. In particular, IPSO can achieve the better quality results on functions F_{11}, and F_{12}. For function F_{10}, IPSO is better than LDIWPSO, AIWPSO, DAPSO, and SSRDIWPSO, but worse than CDIWPSO.

A performance comparison of LDIWPSO, AIWPSO, CDIWPSO, DAPSO, SSRDIWPSO, and IPSO over 30 runs for the average best-so-far solution of functions F_5, F_7, F_9, and F_{11} with dimension 100 is given in Fig. 4. In Fig. 4, compared with the other five algorithms, the convergence speed of IPSO is faster on functions F_5, F_7, F_9, and F_{11}, and IPSO can converge quickly to the optimal solutions on functions F_7, F_9, and F_{11}.

Figure 5 shows the performance comparison of LDIWPSO, AIWPSO, CDIWPSO, DAPSO, SSRDIWPSO, and IPSO over 30 runs for the average best-so-far solution of functions F_1, F_7, F_8, and F_{13} with dimension 1000. In Fig. 5, the convergence speed of IPSO is faster than that of the other five algorithms for functions F_1, F_7, F_8, and F_{13}, and IPSO can converge quickly to the optimal solutions on functions F_1 and F_7.

Now, we evaluate the runtimes of these algorithms under the condition that the convergence accuracy is satisfied. According to the second stopping condition, Tables 9, 10, 11, 12, 13, and 14, respectively, show the average iteration numbers and average runtimes of algorithm for the unimodal and multimodal functions with dimensions 30, 100, and 1000 over 30 runs. In these experiments, the population size is set to 30, the maximum number of iteration is 3000, and the con-



(a)

(b)

(c)

(d)

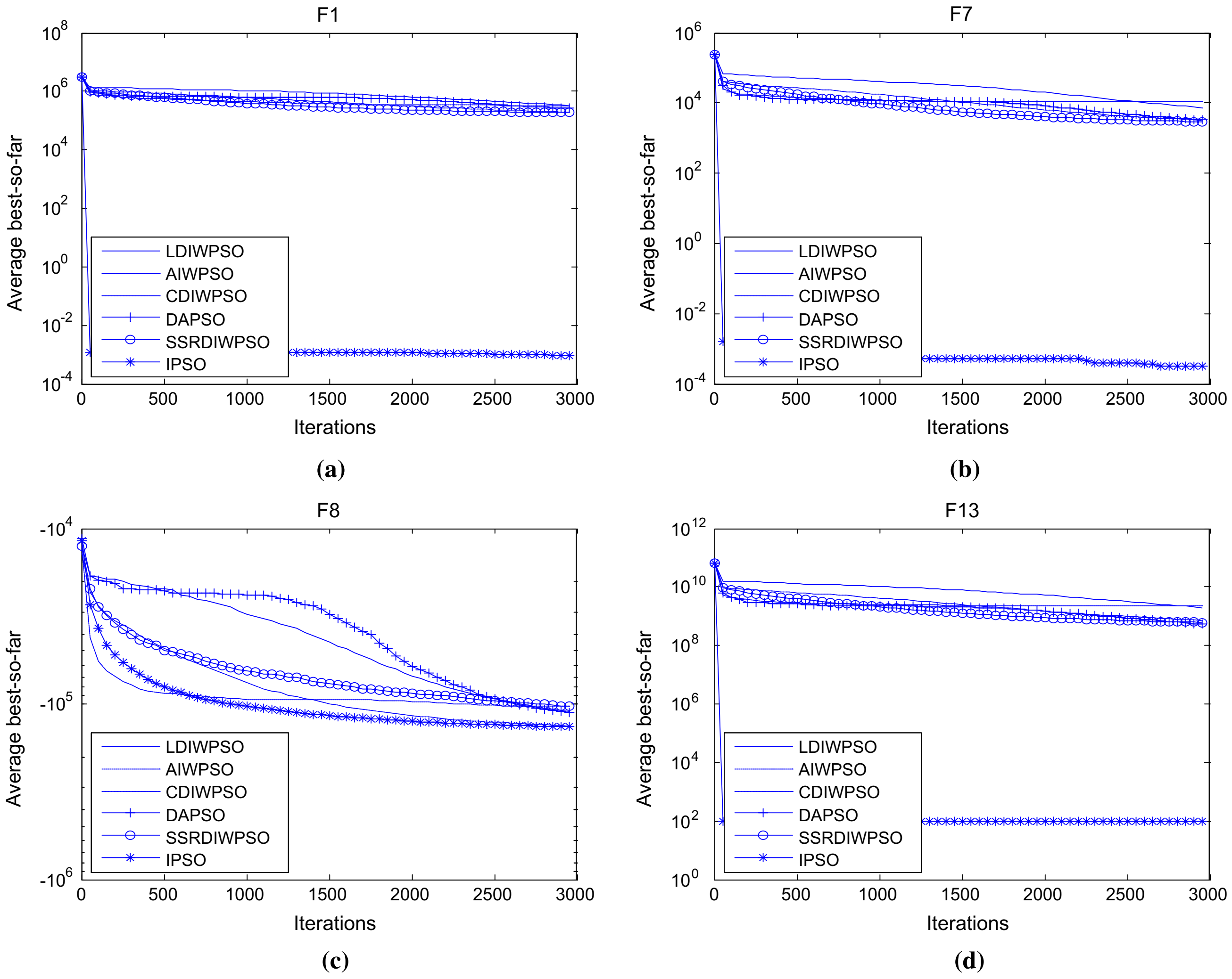
Fig. 4 Performance comparison for functions **a** F_5, **b** F_7, **c** F_9 and **d** F_{11} with $D = 100$ and $K_{\max} = 3000$

vergence accuracy ε is 10^{-5}, 10^{-3}, and 10^{-1} for 30-, 100-, and 1000-dimensional benchmark functions, respectively.

From the average runtimes reported in Tables 9 and 10, for functions F_3, F_6, F_9, F_{11}, and F_{12}, IPSO is better than other five methods. For function F_1, IPSO is slightly better than LDIWPSO and DAPSO, but worse than AIWPSO, CDIWPSO and SSRDPSO. For function 10, except for SSRDIWPSO, IPSO is better than other four algorithms. For functions F_2, F_4, F_5, F_7, F_8 and F_{13}, IPSO is worse than other five algorithms. From the average numbers of iteration given by Tables 9 and 10, before the maximum number of iteration is met, LDIWPSO, AIWPSO, CDIWPSO, DAPSO, and SSRDIWPSO are able to obtain the optimal solutions to achieve the given precision on seven functions, and IPSO can obtain the optimal solutions to achieve the given precision on ten functions, especially, IPSO can converge quickly to the optimal solutions to achieve the given precision on functions F_6 and F_{11}.

From Tables 11 and 12, the average runtimes obtained by LDIWPSO, AIWPSO, CDIWPSO, DAPSO, and SSRDIWPSO are nearly the same for all the benchmark functions in Tables 1 and 2, and the results obtained by IPSO are superior to those obtained by the other five methods on functions F_1, F_3, F_6, F_7, F_{10}, and F_{11}. In addition, average numbers of iteration of IPSO are smaller than those of other five algorithms on eight functions. Specially, IPSO can converge quickly to the optimal solutions to achieve the given precision on functions F_1, F_3, F_6, F_7, and F_{11} before the maximum number of iteration is met, while the average numbers of iteration of the other five algorithms almost reached the maximum number for all functions.

From Tables 13 and 14, the average runtimes obtained by LDIWPSO, AIWPSO, CDIWPSO, DAPSO, and SSRDIWPSO are nearly the same for all benchmark functions, and the results obtained by IPSO are better than those obtained by other five methods on eight functions. For average numbers



(a)

(b)

(c)

(d)

Fig. 5 Performance comparison for functions **a** F_1, **b** F_7, **c** F_8 and **d** F_{13} with $D = 1000$ and $K_{\max} = 3000$

Table 9 Experimental results of benchmark functions in Table 1 with $D = 30$, $\varepsilon = 10^{-5}$, and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_1	Average numbers of iteration	2308	452	1258	2342	666	1552
	Average time	1.6790	0.3146	0.7675	1.6067	0.3599	1.4580
F_2	Average numbers of iteration	2367	619	1297	2382	812	2088
	Average time	1.7884	0.4491	0.8416	1.7004	0.4885	2.0421
F_3	Average numbers of iteration	3000	3000	3000	3000	3000	2007
	Average time	5.6596	5.4673	5.6028	5.7073	5.7573	5.3621
F_4	Average numbers of iteration	3000	3000	3000	3000	3000	3000
	Average time	1.9091	1.8794	1.9258	1.9132	1.9099	2.8663
F_5	Average numbers of iteration	3000	3000	3000	3000	3000	3000
	Average time	2.1324	2.1622	2.1520	2.1633	2.1524	3.5014
F_6	Average numbers of iteration	1468	1342	984	2240	549	11
	Average time	1.3598	1.0085	0.6104	1.6076	0.3830	0.0108
F_7	Std best-so-far	3000	3000	3000	3000	3000	2473
	Average time	2.8927	2.7765	2.8945	2.9170	2.8722	3.8257

Table 10 Experimental results of benchmark functions in Table 2 with $D = 30$, $\varepsilon = 10^{-5}$, and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_8	Average numbers of iteration	3000	3000	3000	3000	3000	3000
	Average time	2.5732	2.9624	2.5754	2.9978	2.1681	3.9155
F_9	Average numbers of iteration	3000	3000	3000	3000	3000	1884
	Average time	2.1565	2.4459	2.1516	2.4872	2.1131	1.9621
F_{10}	Average numbers of iteration	2586	2552	1530	2603	1151	2127
	Average time	2.1688	2.1412	1.1645	2.2470	0.9342	2.1379
F_{11}	Average numbers of iteration	2725	1977	2297	2754	2422	121
	Average time	2.4614	1.9659	2.6101	2.8510	1.7264	0.1570
F_{12}	Average numbers of iteration	2381	2579	2237	2420	2320	1699
	Average time	4.8561	4.1396	4.1333	4.8745	4.1650	4.1058
F_{13}	Average numbers of iteration	2502	2341	1738	2662	1636	2473
	Average time	4.8606	4.6083	3.2949	5.2075	3.0050	6.4416

Table 11 Experimental results of benchmark functions in Table 1 with $D = 100$, $\varepsilon = 10^{-3}$, and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_1	Average numbers of iteration	3000	3000	2999	3000	2573	23
	Average time	2.9538	2.9799	2.9653	3.0237	2.5671	0**.0472**
F_2	Average numbers of iteration	3000	3000	3000	3000	3000	2904
	Average time	3.0671	3.2054	3.1088	3.1367	3.1879	5.1917
F_3	Average numbers of iteration	3000	3000	3000	3000	3000	**26**
	Average time	17.7300	17.8383	17.9115	17.5225	17.5188	**0.2208**
F_4	Average numbers of iteration	3000	3000	3000	3000	3000	3000
	Average time	3.0702	3.0831	3.1041	3.1159	3.0204	5.4742
F_5	Average numbers of iteration	3000	3000	3000	3000	3000	3000
	Average time	3.3060	3.3681	3.3630	3.3623	3.3048	5.9516
F_6	Average numbers of iteration	3000	3000	3000	3000	3000	**12**
	Average time	3.2169	3.2543	3.2050	3.2688	3.1652	**0.0227**
F_7	Average numbers of iteration	3000	3000	3000	3000	3000	115
	Average time	6.1265	6.2055	6.2007	6.2211	6.1102	**0.3564**

Table 12 Experimental results of benchmark functions in Table 2 with $D = 100$, $\varepsilon = 10^{-3}$, and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_8	Average numbers of iteration	3000	3000	3000	3000	3000	3000
	Average time	3.8731	3.9078	5.2376	5.3243	3.8258	8.4320
F_9	Average numbers of iteration	3000	3000	3000	3000	3000	**2664**
	Average time	3.7926	3.7632	3.7978	3.7987	3.6251	5.1857
F_{10}	Average numbers of iteration	3000	3000	3000	3000	3000	2240
	Average time	3.9235	3.9281	3.8767	3.8543	3.7337	**3.6816**
F_{11}	Average numbers of iteration	3000	3000	3000	3000	3000	**27**
	Average time	4.0148	3.9299	5.2297	5.3919	3.6802	**0.0767**
F_{12}	Average numbers of iteration	3000	3000	3000	3000	3000	2808
	Average time	12.1408	12.0672	11.9590	12.1434	11.9910	15.7211
F_{13}	Average numbers of iteration	3000	3000	3000	3000	3000	3000
	Average time	12.0383	12.0222	12.0309	12.1662	11.9261	17.6083

Table 13 Experimental results of benchmark functions in Table 1 with $D = 1000$, $\varepsilon = 10^{-1}$, and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_1	Average numbers of iteration	3000	3000	3000	3000	3000	19
	Average time	46.7564	46.8221	47.0012	46.7230	46.7815	**0.7533**
F_2	Average numbers of iteration	3000	3000	3000	3000	3000	**2904**
	Average time	48.3651	49.7357	48.6913	197.0073	48.7918	60.2335
F_3	Average numbers of iteration	3000	3000	3000	3000	3000	**29**
	Average time	973.0155	1220.0172	975.9041	972.6871	979.0542	**16.0687**
F_4	Average numbers of iteration	3000	3000	3000	3000	3000	17
	Average time	47.4035	47.4089	47.4641	47.5051	47.5802	**0.6650**
F_5	Average numbers of iteration	3000	3000	3000	3000	3000	3000
	Average time	54.0346	53.9481	54.1936	53.1972	54.0624	134.0237
F_6	Average numbers of iteration	3000	3000	3000	3000	3000	**15**
	Average time	48.8854	48.6974	49.1692	48.9833	48.9454	**0.5145**
F_7	Average numbers of iteration	3000	3000	3000	3000	3000	11
	Average time	80.0576	80.2607	80.3156	80.1439	80.2317	**0.5378**

Table 14 Experimental results of benchmark functions in Table 2 with $D = 1000$, $\varepsilon = 10^{-1}$, and $K_{\max} = 3000$

		LDIWPSO	AIWPSO	CDIWPSO	DAPSO	SSRDIWPSO	IPSO
F_8	Average numbers of iteration	3000	3000	3000	3000	3000	3000
	Average time	56.0870	69.2193	69.8211	69.3902	55.7203	135.8571
F_9	Average numbers of iteration	3000	3000	3000	3000	3000	**1207**
	Average time	54.9112	54.8068	55.9542	54.5846	54.4001	**52.5052**
F_{10}	Average numbers of iteration	3000	3000	3000	3000	3000	15
	Average time	57.8708	58.4412	57.7429	57.5804	57.5477	**0.6641**
F_{11}	Average numbers of iteration	3000	3000	3000	3000	3000	**21**
	Average time	61.1641	74.4129	75.4385	74.3132	60.5157	**1.1153**
F_{12}	Average numbers of iteration	3000	3000	3000	3000	3000	**3000**
	Average time	149.0025	148.8029	148.7615	148.1345	149.2124	279.5228
F_{13}	Average numbers of iteration	3000	3000	3000	3000	3000	3000
	Average time	149.7361	149.1231	149.2357	148.9505	149.0678	281.8952

of iteration, IPSO is better than the other five methods on nine functions. In particular, IPSO can converge quickly to the optimal solutions to achieve the given precision on functions F_1, F_3, F_4, F_6, F_7, F_{10}, and F_{11} before the maximum number of iteration is met.

6 Conclusion

The convergence analyses of standard PSO algorithm were studied by some scholars based on the dynamic system theory, difference equation theory, and stochastic process theory, but the studies of convergence in probability are relatively few. The overall goal of this paper is to analyze the convergence of standard PSO algorithm through probability theory. We prove that the standard PSO algorithm is convergence with probability 1 under certain condition. This result is helpful to understand the mechanism of standard PSO algorithm and propose more powerful PSO algorithm. Then, a new improved PSO algorithm that is convergence with probability 1 is proposed. The numerical results obtained from the proposed algorithm demonstrate that the proposed algorithm significantly improves the performance of PSO algorithm in terms of efficiency. In the future, it is very interesting to study probability convergence of other stochastic optimization algorithms.

Acknowledgements This work is partly supported by the National Natural Science Foundation of China (11371071) and Scientific Research Foundation of Liaoning Province Educational Department (L2013426).

Compliance with ethical standards

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

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Adewumi AO, Arasomwan AM (2016) An improved particle swarm optimizer based on swarm success rate for global optimization problems. J Exp Theor Artif Intell 28(3):441–483

Afshar MH (2012) Large scale reservoir operation by constrained particle swarm optimization algorithms. J Hydro-Environ Res 6(1):75–87

Ali MM, Kaelo P (2008) Improved particle swarm algorithms for global optimization. Appl Math Comput 196(2):578–593

Arasomwan AM, Adewumi AO (2014) An investigation into the performance of particle swarm optimization with various chaotic maps. Math Probl Eng pp 1–17, Article ID 178959. doi:10.1155/2014/178959

Behnamian J, Fatemi Ghomi SMT (2010) Development of a PSO-SA hybrid metaheuristic for a new comprehensive regression model to time-series forecasting. Expert Syst Appl 37(2):974–984

Chen CH, Chen YP (2011) Convergence time analysis of particle swarm optimization based on particle interaction. Adv Artif Intell pp. 1–7, Article ID 204750. doi:10.1155/2011/204750

Chuang LY, Yang CH, Li JC (2011) Chaotic maps based on binary particle swarm optimization for feature selection. Appl Soft Comput 11(1):239–248

Clerc M, Kennedy J (2002) The particle swarm: explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evol Comput 6(1):58–73

Ding J, Liu J, Chowdhury KR, Zhang W, Hu Q, Lei J (2014) A particle swarm optimization using local stochastic search and enhancing diversity for continuous optimization. Neurocomputing 137:261–267

Emara HM, Fattah AHA (2004) Continuous swarm optimization technique with stability analysis. In: Proceedings of American control conference, Boston, MA, pp 2811–2817

Feng Y, Teng GF, Wang AX, Yao YM (2007) Chaotic inertia weight in particle swarm optimization. In: Proceedings of the 2nd international conference on innovative computing, information and control, Kumamoto

Fernandez-Martinez JL, Carcia-Gonzalo E, Saraswathi S, Jernigan R, Kloczkowski A (2011) Particle swarm optimization: a powerful family of stochastic optimizers. Analysis, design and application to inverse modelling. In: Proceedings of the 2nd international conference on advances in swarm intelligence. Springer, Berlin, pp 1–8

Franken N, Engelbrecht A (2005) Particle swarm optimization approaches to coevolve strategies for the iterated prisoner's dilemma. IIEEE Trans Evol Comput 9(6):562–579

Gandomi AH, Yun GJ, Yang XS, Talatahari S (2013) Chaos-enhanced accelerated particle swarm optimization. Commun Nonlinear Sci Numer Simul 18(2):327–340

Jiang M, Luo YP, Yang SY (2007) Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm. Inf Process Lett 102(1):8–16

Jin XL, Ma LH, Wu TJ, Qian JX (2007) Convergence analysis of the particle swarm optimization based on stochastic processes. Acta Autom Sin 33(12):1263–1268

Kadirkamanathan V, Selvarajajah K, Fleming PJ (2006) Stability analysis of the particle dynamics in particle swarm optimizer. IEEE Trans Evol Comput 10(3):45–255

Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: IEEE international conference on neural networks, Piscataway, NJ, pp 1942–1948

Li N, Sun DB, Zou T, Qin YQ, Wei Y (2006) An analysis for a particle's trajectory of PSO based on difference equation. Chin J Comput 29(11):2052–2061

Lim WH, Isa NAM (2014) Teaching and peer-learning particle swarm optimization. Appl Soft Comput 18:39–58

Malik RF, Rahman TA, Hashim SZM, Ngah R (2007) New particle swarm optimizer with sigmoid increasing inertia weight. Int J Comput Sci Secur 1(2):35–44

Nickabadi A, Ebadzadeh MM, Safabakhsh R (2011) A novel particle swarm optimization algorithm with adaptive inertia weight. Appl Soft Comput 11:3658–3670

Ozcan, E, Mohan CK (1998) Analysis of a simple particle swarm optimization system. In: Intelligent engineering systems through artificial neural networks, pp 253–258

Ozcan, E, Mohan CK (1999) Particle swarm optimization: surfing the waves. In: Congress on evolutionary computation. IEEE Press, Washington, DC, pp 1939–1944

Pan F, Li XT, Zhou Q, Li WX, Gao Q (2013) Analysis of standord particle swarm optimization algorithm based on Markov chain. Acta Autom Sin 39(4):281–289

Rapaic MR, Kanovic Z (2009) Time-varying PSO-convergence analysis, convergence-related parameterization and new parameter adjustment schemes. Inf Process Lett 109(11):548–552

Shen X, Chi Z, Yang J, Chen C, Chi Z (2010) Particle swarm optimization with dynamic adaptive inertia weight. In: International conference on challenges in environmental science and computer engineering, Wuhan, pp 287–290

Shi YH, Eberhart RC (1999) Empirical study of particle swarm optimization. In: IEEE international conference on evolutionary computation, Washington, DC, pp 1945–1950

Trelea IC (2003) The particle swarm optimization algorithm: convergence analysis and parameter selection. Inf Process Lett 85(6):317–325

van den Bergh F (2002) An analysis of particle swarm optimization. PhD thesis, Department of Computer Science, University of Pretoria

van den Bergh F, Engelbrecht AP (2006) A study of particle swarm optimization particle trajectories. Inf Sci 176(8):937–971

Victoire A, Jeyakumar AE (2005) Reserve constrained dynamic dispatch of units with valve-point effects. IEEE Trans Power Syst 20(3):1273–1282

Wang L, Yang B, Chen Y (2014) Improving particle swarm optimization using multi-layer searching strategy. Inf Sci 274:70–94

Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):81–102

Yu X, Zhang X (2014) Enhanced comprehensive learning particle swarm optimization. Appl Math Comput 242:265–276

Zou F, Wang L, Hei X, Chen D, Yang D (2014) Teaching-learning-based optimization with dynamic group strategy for global optimization. Inf Sci 273:112–131