Swarm and Evolutionary Computation 99 (2025) 102138

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

A multiple direction search algorithm for continuous optimization ☆

Wei Huang [a], Jun He [b],*, Liehuang Zhu [a]

[a] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China
[b] Department of Computer Science, Nottingham Trent University, Nottingham, NG11 8NS, UK

ARTICLE INFO

Keywords:
Continuous optimization
Evolutionary algorithm
Particle swarm optimization
Principal component analysis
Convergence rate

ABSTRACT

The particle swarm optimization algorithm has been successfully applied to various optimization problems. One of its key features is the combination of particle velocity and search direction towards the optimal position in the history and swarm. Recognizing the limitations of the particle swarm optimization algorithm, this paper proposes a new evolutionary algorithm called the multiple direction search algorithm. The algorithm integrates five different search directions, including a multi-point direction constructed using principal component analysis. The integrated direction is generated by the weighted sum of the search directions. Theoretical analysis shows that under mild conditions, the rate of convergence along the weighted direction is no worse than the rate of convergence along the best of single search directions by a positive constant, or even faster in certain cases. The performance of the proposed algorithm was evaluated on three benchmark test suites by computer simulation. Experimental results demonstrate that the proposed method outperforms seven state-of-the-art particle swarm optimization algorithms.

1. Introduction

Particle swarm optimization (PSO), inspired by social behavior in the bird flock or fish school, is a kind of population-based evolutionary algorithms (EAs) originally for continuous optimization. Currently PSO has been applied to both continuous and combinatorial optimization problems such as single-objective optimization [1], multimodal optimization [2], many-objective optimization [3], multiobjective combinatorial optimization [4], large-scale optimization [5], high-dimensional expensive problems [6], expensive constrained multimodal problems [7].

From the viewpoint of search, standard PSO combines the velocity of a particle with search directions towards the personal best position of the particle and the global best position of a group. Then the new velocity is updated through combining them together through the weighted sum method.

Several limitations exist in PSO, such as swarm explosion, local convergence and transformation invariance [8]. Aware of these limitations, the current paper proposes a new EA called the multiple direction search algorithm (DSA). Similar to PSO, DSA is designed as a population-based evolutionary algorithm without crossover. In each generation, particles construct multiple different search directions, then search along an integrated direction and generate offspring. But unlike PSO, particles have no velocity and only move to a better position.

The main research hypothesis of this paper is that using multiple search directions can outperform using a single search direction. There are two research questions in this paper.

1. From a theoretical perspective, can it be proven that a DSA using multiple search directions performs better than an algorithm using the single search direction?
2. From a computer simulation perspective, can a DSA be designed and implemented that outperforms other state-of-the-art PSO algorithms?

To address the first question, we provide a theoretical analysis of DSA using the convergence rate of EAs. The analysis employs the convergence in mean of the error sequence [9], which is commonly used in experimental studies, unlike the convergence in probability favored in theoretical studies [8]. The one-step convergence rate [9] is utilized to establish theoretical results. An encouraging finding is that even if the convergence rate along any single search direction is slow, DSA can still achieve fast convergence along the integrated direction in certain cases.

To address the second question, we designed a DSA incorporating five search directions. To validate our theory, we conducted computational simulations on three benchmark test suites. The experiments

☆ This work was supported by the National Major Scientific Instruments and Equipments Development Project of National Natural Science Foundation of China under Grant 62227805.

* Corresponding author.
E-mail address: jun.he@ntu.ac.uk (J. He).

https://doi.org/10.1016/j.swevo.2025.102138
Received 26 June 2023; Received in revised form 20 July 2025; Accepted 17 August 2025
Available online 10 September 2025

confirmed that even if one single-point direction appears “useless” (as they fail to generate a new child), the DSA integrating multiple directions remains effective.

The remainder of this paper is organized as follows: Section 2 provides a review of search directions in PSO and highlights its limitations. Section 3 discusses the benefits of employing multiple search directions, with a focus on convergence rate improvements. Section 4 details the design of DSA, which incorporates five distinct search directions. Section 5 presents simulation results based on three benchmark suites. Finally, Section 6 concludes the paper.

2. Literature survey

This section focuses on briefly reviewing the search directions used in PSO as they are directly related to DSA and discussing the limitations of existing PSO algorithms. The original PSO [10] applies a formula to update its velocity. Given particle i and its current velocity $\mathbf{v}_t^i \in \mathbb{R}^n$ and position $\mathbf{x}_t^i \in \mathbb{R}^n$ at time t, the velocity at time $t+1$ is

$$\mathbf{v}_{t+1}^i = \mathbf{v}_t^i + \phi_1 \mathbf{R}_{t1}^i (\mathbf{p}_t^i - \mathbf{x}_t^i) + \phi_2 \mathbf{R}_{t2}^i (\mathbf{g}_t - \mathbf{x}_t^i) \tag{1}$$

where $\phi_1, \phi_1 \in \mathbb{R}$ are cognitive and social acceleration weights, respectively. The matrices $\mathbf{R}_1, \mathbf{R}_2 \in \mathbb{R}^{n\times n}$ take random values uniformly distributed in the range $[0,1]$. Two search directions are introduced in the formula: the cognitive influence $\mathbf{p}^i - \mathbf{x}^i$ on the individual best position; and the social influence $\mathbf{g} - \mathbf{x}^i$ on the global best position.

In the standard PSO, an inertia weight ω [11] is added to control the contribution of $\mathbf{v}_t^i$.

$$\mathbf{v}_{t+1}^i = \omega\mathbf{v}_t^i + \phi_1 \mathbf{R}_{t1}^i (\mathbf{p}_t^i - \mathbf{x}_t^i) + \phi_2 \mathbf{R}_{t2}^i (\mathbf{g}_t - \mathbf{x}_t^i) \tag{2}$$

A constriction factor χ [12] is added to ensure better local convergence of PSO.

$$\mathbf{v}_{t+1}^i = \chi[\omega\mathbf{v}_t^i + \phi_1 \mathbf{R}_{t1}^i (\mathbf{p}_t^i - \mathbf{x}_t^i) + \phi_2 \mathbf{R}_{t2}^i (\mathbf{g}_t - \mathbf{x}_t^i)] \tag{3}$$

PSO variants were designed through modifying the formula of updating velocity in standard PSO. For the Standard PSO algorithm [13], the search direction is constructed using a spherical distribution. In LcRiPSO [14], an operator applies perturbation on $\mathbf{p}$ to improve local convergence.

Besides modifying the formula of updating velocity, search directions may be changed using some strategy or new search directions are created. In the prey–predator PSO [15], slothful particles are deleted or transformed with a reassigned velocity. The FES-assisted PSO [16] has a new direction towards the global best position obtained by the RBF-assisted. In AGLDPSO [5], a population is split into subpopulation and search directions are different in subpopulations. These results show using more search directions may improve the performance of PSO.

Standard PSO has several limitations, such as swarm explosion, local convergence and transformation invariance [8]. One limitation in standard PSO is swarm explosion [8,12]. The velocity of particles could go towards infinity for some values of acceleration and inertia coefficients. Solutions were proposed such as the restriction of the position of a particle or its velocity within a fixed range [17]. The problem leads to stability analysis of PSO [13,18]. To tackle swarm explosion, the proposed DSA adopts a simple solution, that is to drop the velocity of a particle, but only keep search directions. The search is always restricted within a reasonable area covered by a population.

An inherent limitation in standard PSO is the lack of local convergence [8]. Because a particle has a velocity, it always can move to a new position. A solution is to add a constriction factor [12]. The proposed DSA drops the velocity of particles and takes elitist selection to update the position of a particle. Thus, local convergence is not a problem in DSA.

Another limitation is linked to rotation invariance [8]. Although standard PSO performs well on separable functions, it was reported [19] that its performance on non-separable functions is worse than Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) and Differential Evolution (DE). To handle this issue, a solution is a user-definable function in the velocity update rule which increases perturbation and prevent stagnation [14]. Inspired from CMA-ES, the proposed DSA provides an alternative solution. Principal component analysis is applied to generating principal components and first principle component is taken as a multi-point search direction. DSA also uses a non-optimal search direction from the current particle towards the farthest particle in a population.

3. The advantage of using multiple search directions

This section explains the advantage of using multiple search directions from the perspective of the convergence rate.

3.1. One-step convergence rate

Consider a minimization problem: $\min f(\mathbf{x}), \mathbf{x} \in \mathbb{R}^n$ where n is the dimension. Given a search direction $\mathbf{v}$, a child $\mathbf{y}$ is generated by

$$\mathbf{y} = \mathbf{x} + \sigma\mathbf{v},$$

where σ is the step size. Let $e(\mathbf{x}) = |f(\mathbf{x}) - f_{\min}|$ denote the error between $f(\mathbf{x})$ and the minimal function value $f_{\min}$. Then the one-step convergence rate is defined as follows [9].

Definition 1. Let $r(\mathbf{x}, \mathbf{v})$ denote the *one-step convergence rate* starting from $\mathbf{x}$ along a search direction $\mathbf{v}$, defined as

$$r(\mathbf{x}, \mathbf{v}) = \frac{e(\mathbf{x}) - e(\mathbf{y})}{e(\mathbf{x})} = \frac{e(\mathbf{x}) - e(\mathbf{x} + \sigma\mathbf{v})}{e(\mathbf{x})}. \tag{4}$$

For ease of analysis, we only consider elitist selection, i.e., the particle only moves to a new position $\mathbf{y}$ where $f(\mathbf{y}) \le f(\mathbf{x})$. In this case, the convergence rate is never negative. The convergence rate depends on the search space dimension n, therefore, it is a function of n.

Definition 2. The convergence rate $r(\mathbf{x}, \mathbf{v}_1)$ along the direction $\mathbf{v}_1$ is *asymptotically no worse*[1] than the convergence rate $r(\mathbf{x}, \mathbf{v}_2)$ along another direction $\mathbf{v}_2$ if

$$\frac{r(\mathbf{x}, \mathbf{v}_1)}{r(\mathbf{x}, \mathbf{v}_2)} = \Omega(1). \tag{5}$$

Furthermore, the convergence rate $r(\mathbf{x}, \mathbf{v}_1)$ is *significantly faster* than the convergence rate $r(\mathbf{x}, \mathbf{v}_2)$ if

$$\frac{r(\mathbf{x}, \mathbf{v}_1)}{r(\mathbf{x}, \mathbf{v}_2)} = \omega(1). \tag{6}$$

Eq. (5) implies that $r(\mathbf{x}, \mathbf{v}_1)$ and $r(\mathbf{x}, \mathbf{v}_2)$ differ by a positive constant, although it does not specify which rate is larger. In contrast, Eq. (6) explicitly asserts that $r(\mathbf{x}, \mathbf{v}_1)$ is considerably larger than $r(\mathbf{x}, \mathbf{v}_2)$.

3.2. Weighted sum of multiple search directions

Given several search directions $\mathbf{v}_1, \dots, \mathbf{v}_k$, an integrated search direction can be constructed by taking a weighted sum of these directions as follows:

$$\mathbf{v}_w = \sum_{i=1}^{k} w_i \mathbf{v}_i, \qquad \text{subject to } \sum_i w_i = 1. \tag{7}$$

The weights $w_i \ge 0$ are randomly selected to control the contribution of the search direction $\mathbf{v}_i$. Let $\mathbf{W} = \{\mathbf{w} = (w_1, \dots, w_k); \sum_i w_i = 1\}$ denote

[1] Let $r(n)$ be a function of n. The asymptotic notation $r(n) = \Omega(1)$ means that there exists a constant $C > 0$ such that $r(n) \ge C$ for all sufficiently large n. The notation $r(n) = \omega(1)$ means that $\lim_{n\to+\infty} r(n) = +\infty$.

the set of all potential weights, and $p(\mathbf{w})$ the joint probability density function of $\mathbf{w}$. The expected convergence rate is

$$r(\mathbf{x}, \mathbf{v}_w) = \frac{\int_{\mathbf{w}\in\mathbf{W}}[e(\mathbf{x}) - e(\mathbf{x} + \sigma\mathbf{v}_w)p(\mathbf{w})]d\mathbf{w}}{e(\mathbf{x})}. \tag{8}$$

The following theorem compares the convergence rate obtained by searching a weighted sum of directions with the convergence rate obtained by using the best single direction. Although the theorem is formulated for two directions, the principle can be extended to apply to multiple directions.

Theorem 1. *Let $\mathbf{v}_1$ and $\mathbf{v}_2$ be two search directions. Suppose the following conditions are satisfied:*

1. *For any weights $(w_1, w_2) \in [1-\epsilon, 1] \times [0, \epsilon]$ with $\epsilon > 0$, the convergence rate along the weighted direction $\mathbf{v}_w$ satisfies*
   $$\frac{r(\mathbf{x}, \mathbf{v}_w)}{r(\mathbf{x}, \mathbf{v}_1)} = \Omega(1).$$
   The probability of selecting weights $(w_1, w_2) \in [1-\epsilon, 1] \times [0, \epsilon]$ is $\Omega(1)$.
2. *For any weights $(w_1, w_2) \in [0, \epsilon] \times [1-\epsilon, 1]$ with $\epsilon > 0$, the convergence rate along the weighted direction $\mathbf{v}_w$ satisfies*
   $$\frac{r(\mathbf{x}, \mathbf{v}_w)}{r(\mathbf{x}, \mathbf{v}_2)} = \Omega(1).$$
   The probability of selecting weights $(w_1, w_2) \in [0, \epsilon] \times [1-\epsilon, 1]$ is $\Omega(1)$.

Then, the convergence rate along the weighted direction $\mathbf{v}_w$ satisfies

$$\frac{r(\mathbf{x}, \mathbf{v}_w)}{\max\left\{r(\mathbf{x}, \mathbf{v}_1),\, r(\mathbf{x}, \mathbf{v}_2)\right\}} = \Omega(1).$$

Proof. From (8) and the conditions of the theorem, we have

$$\begin{aligned}
r(\mathbf{x}, \mathbf{v}_w) &= \frac{\int_{\mathbf{w}\in\mathbf{W}}\left[e(\mathbf{x}) - e(\mathbf{x} + \sigma\sum_i w_i\mathbf{v}_i)p(\mathbf{w})\right]d\mathbf{w}}{e(\mathbf{x})} \\
&\geq \frac{\left(\int_{\mathbf{w}\in[1-\epsilon,1]\times[0,\epsilon]} + \int_{\mathbf{w}\in[0,\epsilon]\times[1-\epsilon,1]}\right)\left[e(\mathbf{x}) - e(\mathbf{x} + \sigma\sum_i w_i\mathbf{v}_i)p(\mathbf{w})\right]d\mathbf{w}}{e(\mathbf{x})} \\
&\geq \Pr(\mathbf{w} \in [1-\epsilon, 1] \times [0, \epsilon])\,\Omega(1)\,r(\mathbf{x}, \mathbf{v}_1) \\
&\quad + \Pr(\mathbf{w} \in [0, \epsilon] \times [1-\epsilon, 1])\,\Omega(1)\,r(\mathbf{x}, \mathbf{v}_2) \\
&\geq \Omega(1)\,\max\left\{r(\mathbf{x}, \mathbf{v}_1), r(\mathbf{x}, \mathbf{v}_2)\right\}.
\end{aligned}$$

This proves the conclusion. □

The above theorem indicates that, under Conditions (1) and (2), the convergence rate along the weighted sum direction is no worse than that of the most effective single search direction by a constant $C > 0$. Conditions (1) and (2) are mild and can be easily satisfied in population-based algorithms. Consider a population of N particles, and assign a search direction to the ith particle, defined as

$$\mathbf{v}_i = \frac{N-i}{N-1}\mathbf{v}_1 + \frac{i-1}{N-1}\mathbf{v}_2, \quad \text{for } i = 1, \dots, N.$$

Under this scheme, the first particle always follows the direction $\mathbf{v}_1$, while the Nth particle always follows the direction $\mathbf{v}_2$. The remaining particles interpolate between these two directions to ensure the diversity of search directions. In this case, the constant $C \geq 1$.

In addition, since the weighted sum direction $\mathbf{v}$ can be a new direction different from the existing directions $\mathbf{v}_1, \dots, \mathbf{v}_k$, the convergence rate along the weighted sum direction may be significantly faster than the convergence rate along the best of single directions in some cases. The following theorem is proved for two directions, but the conclusion can be extended to multiple directions.

Theorem 2. *Let $\mathbf{v}_1$ and $\mathbf{v}_2$ be two search directions. Suppose the following conditions are satisfied:*

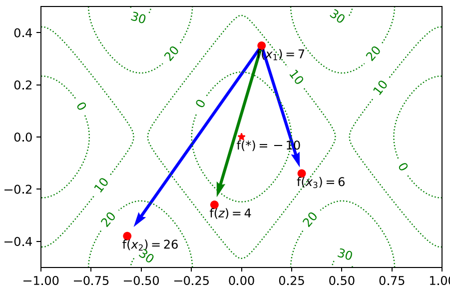

Fig. 1. The Rastrigin function.

1. *For any weight $(w_1, w_2) \in [\mu_1, \lambda_1] \times [\mu_2, \lambda_2]$ with $0 < \mu_i < \lambda_i < 1$, the convergence rate along the weighted direction $\mathbf{v}_w$ satisfies*
   $$\frac{r(\mathbf{x}, \mathbf{v}_w)}{\max\{r(\mathbf{x}, \mathbf{v}_1), r(\mathbf{x}, \mathbf{v}_2)\}} = \omega(1).$$
2. *The probability of selecting weights $(w_1, w_2) \in [\mu_1, \lambda_1] \times [\mu_2, \lambda_2]$ is $\Omega(1)$.*

Then, the convergence rate along the weighted direction $\mathbf{v}_w$ satisfies

$$\frac{r(\mathbf{x}, \mathbf{v}_w)}{\max\left\{r(\mathbf{x}, \mathbf{v}_1),\, r(\mathbf{x}, \mathbf{v}_2)\right\}} = \omega(1).$$

Proof. From (8), it has

$$\begin{aligned}
r(\mathbf{x}, \mathbf{v}) &= \frac{\int_{\mathbf{w}\in\mathbf{W}}[e(\mathbf{x}) - e(\mathbf{x} + \sum_i w_i\mathbf{v}_i)p(\mathbf{w})]d\mathbf{w}}{e(\mathbf{x})} \\
&\geq \frac{\int_{\mathbf{w}\in[\mu_1,\lambda_1]\times[\mu_2,\lambda_2]}[e(\mathbf{x}) - e(\mathbf{x} + \sum_i w_i\mathbf{v}_i)p(\mathbf{w})]d\mathbf{w}}{e(\mathbf{x})} \\
&= \Pr(\mathbf{w} \in [\mu_1, \lambda_1] \times [\mu_2, \lambda_2])\,\omega(1)\,\max\{r(\mathbf{x}, \mathbf{v}_1), r(\mathbf{x}, \mathbf{v}_2)\} \\
&= \omega(1)\max\{r(\mathbf{x}, \mathbf{v}_1), r(\mathbf{x}, \mathbf{v}_2)\}.
\end{aligned}$$

This proves the conclusion. □

Condition (1) of the theorem implies that there exists a region of weights $\mathbf{w} \in [\mu_1, \lambda_1] \times [\mu_2, \lambda_2]$ such that convergence along the weighted sum direction is significantly faster than convergence along $\mathbf{v}_1$ or $\mathbf{v}_2$. This condition is strong and only holds in some cases. Condition (2) of the theorem requires that the probability of weight selection within this region is high enough. This theorem highlights the potential advantage of using a weighted sum of search directions, which in some cases can significantly speed up convergence compared to relying solely on a single search direction.

It is worth mentioning that the optimal weight vector satisfies:

$$\mathbf{w}_{\text{opt}} = \arg\max\{r(\mathbf{x}, \mathbf{v}_\mathbf{w}), \mathbf{w} \in \mathbf{W}\}. \tag{9}$$

Since the weight set $\mathbf{W}$ is closed, it is guaranteed to have a maximum value. In general, the optimal weight vector is unknown. Therefore, one strategy is to adjust the weights adaptively, but this is beyond the scope of this paper.

3.3. An illustrative example

Consider the problem of minimizing the Rastrigin function. Fig. 1 illustrates the function, where the optimal point is $[0, 0]$ and the optimal function value is -10.

$$f(\mathbf{x}) = x^2 + y^2 - 10\cos(2\pi x) - 10\cos(2\pi y) + 10, \quad \mathbf{x} \in [-1, 1] \times [-0.5, 5]. \tag{10}$$

Suppose we have two search directions as shown in the figure:

$$\mathbf{v}_2 = \mathbf{x}_2 - \mathbf{x}_1, \quad \text{and} \quad \mathbf{v}_3 = \mathbf{x}_3 - \mathbf{x}_1.$$

The direction $\mathbf{v}_2$ corresponds to the uphill direction of the objective function because $f(\mathbf{x}_2) = 26 > f(\mathbf{x}_1) = 7$. Therefore, the particle does not move from $\mathbf{x}_1$ to the position $\mathbf{x}_2$, but stays at position $\mathbf{x}_1$. The corresponding convergence rate along $\mathbf{v}_2$ is 0.

In contrast, the direction $\mathbf{v}_3$ yields a downhill descent, since $f(\mathbf{x}_3) = 6 < f(\mathbf{x}_1) = 7$. Therefore, the particle moves to the new position $\mathbf{x}_3$ and has a convergence rate of

$$r(\mathbf{x}_1, \mathbf{v}_3) = \frac{|f(\mathbf{x}_1) - f(\mathbf{x}_{\text{opt}})| - |f(\mathbf{x}_3) - f(\mathbf{x}_{\text{opt}})|}{|f(\mathbf{x}_1) - f(\mathbf{x}_{\text{opt}})|} = \frac{7-6}{7+10} = \frac{1}{17}.$$

If a particle chooses the weight $w_1 = 1$ with probability $9/10$, then the rate of convergence along $\mathbf{v}_3$ is at least

$$r(\mathbf{x}_1, \mathbf{v}_3) = \frac{9}{10}\frac{|f(\mathbf{x}_1) - f(\mathbf{x}_{\text{opt}})| - |f(\mathbf{x}_3) - f(\mathbf{x}_{\text{opt}})|}{|f(\mathbf{x}_1) - f(\mathbf{x}_{\text{opt}})|} = \frac{9}{170} < \frac{1}{17}.$$

Theorem 1 generalizes this observation.

Now we construct a new weighted sum direction using equal weights:

$$\mathbf{v}_w = \frac{1}{2}(\mathbf{x}_2 - \mathbf{x}_1) + \frac{1}{2}(\mathbf{x}_3 - \mathbf{x}_1).$$

Evaluating the function at the resulting point $\mathbf{z}$ yields $f(\mathbf{z}) = 4 < f(\mathbf{x}_1) = 7$, indicating a stronger descent and a faster convergence rate:

$$r(\mathbf{x}_1, \mathbf{v}_w) = \frac{|f(\mathbf{x}_1) - f(\mathbf{x}_{\text{opt}})| - |f(\mathbf{z}) - f(\mathbf{x}_{\text{opt}})|}{|f(\mathbf{x}_1) - f(\mathbf{x}_{\text{opt}})|} = \frac{7-4}{7+10} = \frac{3}{17} > \frac{1}{17}.$$

Theorem 2 generalizes this observation.

4. A five-direction search algorithm

The key point in DSA is to design multiple search directions. This section describes the construction of a five-direction search algorithm.

4.1. A simple multiple direction search algorithm

Consider a minimization problem: $\min f(\mathbf{x}), \mathbf{x} \in \mathbb{R}^n$, where f is called a fitness function. The smaller f value, the better fitness in minimization. The pseudo code of the DSA algorithm is described in Algorithm 1. For each generation, the time complexity of DSA is $T_{\max} \times (5 + k)$ where k denotes the number of directions.

Algorithm 1 A Simple Multiple Direction Search Algorithm (DSA)

```
Randomly generate an initial population X_0.
Calculate the fitness value of each particle in the population.
for t = 0, ..., T_max do
    Randomly select a particle x in the population X_t.
    Construct k search directions v_1, ..., v_k.
    Generate an integrated search direction v.
    Search along the direction v and generate a child y.
    Calculate the fitness value of y.
    Generate next population X_{t+1} by removing the worst particle
    from X_t ∪ {y}.
end for
```

Given a particle $\mathbf{x}$, a child $\mathbf{y}$ is generated by

$$\mathbf{y} = \mathbf{x} + \sigma \mathbf{v} \tag{11}$$

where the vector $\mathbf{v} \in \mathbb{R}^n$ represents a search direction. $\sigma \in \mathbb{R}$ is a step size. Selection is elitist because the best particle in population X_t is always preserved.

4.2. Multiple search directions and integrated direction

The general process of constructing a search direction is divided into two steps: given a particle $\mathbf{x}$,

1. select one or more particles other than $\mathbf{x}$ from the population;
2. use the information from these particles (with or without $\mathbf{x}$) to construct the search direction.

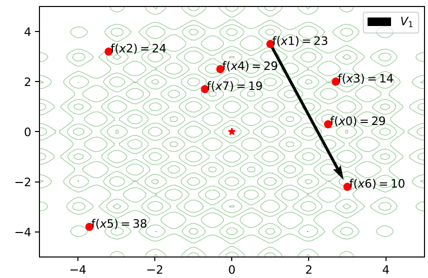

Fig. 2. An example of steepest-descent single-point direction: $\mathbf{x}_6 - \mathbf{x}_1$. The notation $\star$ represents the minimum of $f(\mathbf{x})$.

Five different search directions are constructed in this paper, namely

1. steepest-descent single-point direction $\mathbf{v}_1$;
2. farthest-distance single-point direction $\mathbf{v}_2$;
3. hybrid two-point direction $\mathbf{v}_3$;
4. random two-point direction $\mathbf{v}_4$;
5. multi-point direction $\mathbf{v}_5$.

The weighted sum is used to generate the combined direction, that is, summing the five directions:

$$\mathbf{v} = w_1\mathbf{v}_1 + \cdots + w_5\mathbf{v}_5. \tag{12}$$

A simple implementation of the weighted sum is to assign equal random weights, such that their expected value $\mathrm{E}[w_1] = \cdots = \mathrm{E}[w_5]$, and the sum of the weights is normalized to 1.

4.3. Steepest-descent and farthest-distance single-point directions

The single-point direction involves selecting one particle (point) other than the current particle $\mathbf{x}$ to construct a direction. The steepest-descent single-point direction is defined from $\mathbf{x}$ towards the best particle $\mathbf{x}^*$ in the current population X, as follows:

$$\mathbf{v}_1 = \begin{cases} \mathbf{x}^* - \mathbf{x} & \text{if } \mathbf{x} \neq \mathbf{x}^*, \\ 0 & \text{else,} \end{cases} \tag{13}$$

where the best particle $\mathbf{x}^* = \arg\min\{f(\mathbf{y}); \mathbf{y} \in X\}$. If there are multiple best particles, $\mathbf{x}^*$ is selected randomly. This direction aims to exploit the area around the best particle in the population. Fig. 2 shows an example of the steepest-descent single-point direction.

The farthest-distance single-point direction is towards the $\mathbf{x}^\sharp$ which is the farthest particle away from $\mathbf{x}$ under the Euclidean distance.

$$\mathbf{v}_2 = \begin{cases} \mathbf{x}^\sharp - \mathbf{x} & \text{if } \mathbf{x} \neq \mathbf{x}^\sharp, \\ 0 & \text{else.} \end{cases} \tag{14}$$

where the farthest particle $\mathbf{x}^\sharp = \arg\max\{\|\mathbf{y} - \mathbf{x}\|_2; \mathbf{y} \in X\}$. This direction aims to increase the search range. Fig. 3 shows an example of the farthest-distance single-point direction.

Using only a single search direction (either $\mathbf{v}_1$ or $\mathbf{v}_2$) cannot generate any new child. In fact, the child $\mathbf{y}$ is the same as either $\mathbf{x}^*$ or $\mathbf{x}^\sharp$.

$$\mathbf{y} = \mathbf{x} + \mathbf{v}_1 = \mathbf{x} + \mathbf{x}^* - \mathbf{x} = \mathbf{x}^*. \tag{15}$$

$$\mathbf{y} = \mathbf{x} + \mathbf{v}_2 = \mathbf{x} + \mathbf{x}^\sharp - \mathbf{x} = \mathbf{x}^\sharp. \tag{16}$$

Although the above single-point search directions do not generate a new child, it still contributes to the integrated search direction. As proven in Section 3, the integrated direction can still perform effectively. This is an advantage of DSA.

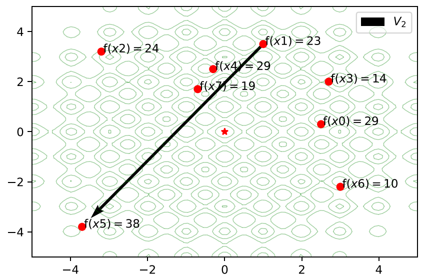

Fig. 3. An example of the farthest-distance single-point direction: $\mathbf{x}_5 - \mathbf{x}_1$.

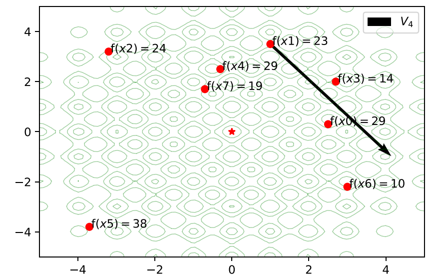

Fig. 5. An example of random descent two-points search direction $\mathbf{x}_6 - \mathbf{x}_7$.

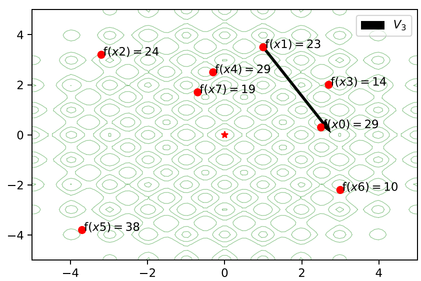

Fig. 4. An example of the steepest-descent two-points direction: $(\mathbf{x}_6+\mathbf{x}_3)/2-\mathbf{x}_1$.

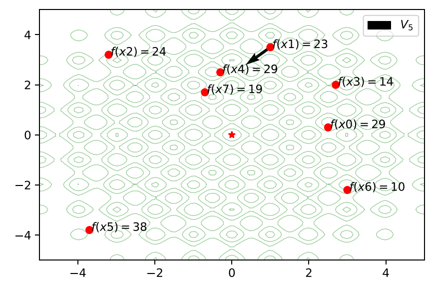

Fig. 6. An example of a multiple-point direction: $\mathbf{v}_5 \leftarrow PCA(\mathbf{x}_0, \dots, \mathbf{x}_6)$.

4.4. Steepest-descent and random-descent two-point directions

The steepest-descent two-point direction is from the current particle $\mathbf{x}$ to the center of the best two particles (other than $\mathbf{x}$) in the current population. Its procedure includes two steps:

1. find the best particle

   $\mathbf{x}_1 = \arg\min\{f(\mathbf{y}); \mathbf{y} \in X \setminus \{\mathbf{x}\}\}$

   and the second-best particle

   $\mathbf{x}_2 = \arg\min\{f(\mathbf{y}); \mathbf{y} \in X \setminus \{\mathbf{x}, \mathbf{x}_1\}\}$.
2. Construct a search direction, given by

$$\mathbf{v}_3 = \begin{cases} \frac{\mathbf{x}_1+\mathbf{x}_2}{2} - \mathbf{x}, & \text{if } f(\mathbf{x}_1) \le f(\mathbf{x}_2) \le f(\mathbf{x}), \\ \mathbf{x}_1 - \mathbf{x}, & \text{if } f(\mathbf{x}_1) \le f(\mathbf{x}) < f(\mathbf{x}_2), \\ 0, & \text{else.} \end{cases}$$

Fig. 4 illustrates an example of steepest-descent two-points search direction. This direction aims at exploiting the neighborhood of the best two particles.

The random descent two-point direction is to construct a direction between two particles (other than $\mathbf{x}$) whose fitness values are better than $f(\mathbf{x})$. Its procedure includes two steps:

1. construct the better set

   $S_{\text{bet}} = \{\mathbf{y} \in X; f(\mathbf{y}) < f(\mathbf{x})\}$.
2. randomly select two particles $\mathbf{x}_1, \mathbf{x}_2$ from S_{bet}.
3. construct a search direction by

$$\mathbf{v}_4 = \begin{cases} \mathbf{x}_1 - \mathbf{x}_2 & \text{if } f(\mathbf{x}_1) \le f(\mathbf{x}_2) < f(\mathbf{x}), \\ 0 & \text{else} \end{cases} \tag{17}$$

Fig. 5 describes an example of random descent two-points search direction. Note that this direction is not from the current particle $\mathbf{x}$ to other existing particles. Instead, it is a direction between two better particles. It is similar to mutation in DE, but in DSA, two selected particles are better than the current particle and the direction is from a worse particle to a better one. This direction aims to increase direction diversity.

4.5. Multi-point search direction

The multi-point direction is constructed from three or more particles from the current population. To extract information of multiple particles, PCA is used. The first principle component found in PCA is taken as the search direction. The procedure of multi-point direction is described as follows:

1. Select m particles $\mathbf{x}_1, \dots, \mathbf{x}_m$ from the population. Each particle $\mathbf{x}_i$ represented by a n-dimensional vector $(\mathbf{x}_{i1}, \dots \mathbf{x}_{in})$. The selected m particles are written in a $m \times n$ matrix $\mathbf{A}$:

$$\begin{pmatrix} \mathbf{x}_{11} & \mathbf{x}_{12} & \cdots & \mathbf{x}_{1n} \\ \mathbf{x}_{21} & \mathbf{x}_{22} & \cdots & \mathbf{x}_{2n} \\ \vdots & \vdots & & \vdots \\ \mathbf{x}_{m1} & \mathbf{x}_{m2} & \cdots & \mathbf{x}_{mn} \end{pmatrix}. \tag{18}$$

2. Calculate the mean value $\mathbf{b}$ of $\mathbf{x}_1, \dots, \mathbf{x}_m$ which is written in a n-dimensional vector:

$$\mathbf{b} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i. \tag{19}$$

3. Calculate the covariance matrix $\mathbf{C}$ which is a $n \times n$ matrix:

$$\mathbf{C} = \frac{1}{m-1} \sum_{i=1}^{m} (\mathbf{x}_i - \mathbf{b})(\mathbf{x}_i - \mathbf{b})^T. \tag{20}$$

4. Calculate the eigenvectors $\mathbf{u}_1, \dots, \mathbf{u}_n$ of the covariance matrix $\mathbf{C}$. Select the first primary component $\mathbf{u}_1$ (the eigenvector with the largest eigenvalue) as the search direction $\mathbf{v}_5$.

Fig. 6 illustrates an example of a multi-point direction. Here, PCA is employed to generate a search direction, which is analogous to mutation. This differs from the PCA-projection in [20], where PCA is used to project particles (points) onto new positions, working similarly to crossover.

Finally, Fig. 7 shows an example of five search directions and one integrated search direction. The figure shows that different directions have different search ranges, where the single-point direction with the farthest distance makes the search range farther, while the multi-point direction focuses on local search.

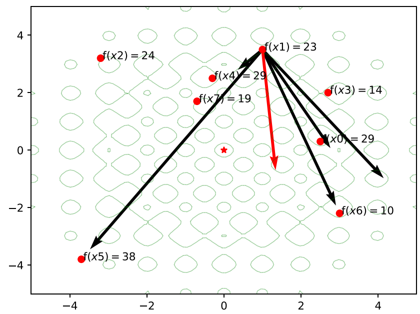

Fig. 7. An example of five search directions and one integrated search direction.

Table 1
Maximal number of function evaluations.

F	DSA/DE/CMA-ES/GA/PSO/GSO	CEP/FEP/CES/FES
F1	150 000	150 000
F2	150 000	200 000
F3	250 000	500 000
F4	150 000	500 000
F5	150 000	2 000 000
F6	150 000	150 000
F7	150 000	300 000
F8	150 000	900 000
F9	250 000	500 000
F10	150 000	150 000
F11	150 000	200 000

5. Experiments and results

This section reports computer experiments and results. The performance of DSA is evaluated on different benchmark suites and compared with eleven classical EAs, seven state-of-the-art PSO variants and also five leading EAs in the CEC 2022 single-objective bounded constrained numerical optimization competition.

5.1. Evaluation on classical test functions and comparison with classical EAs

5.1.1. Experimental setting

A set of thirteen unimodal and multimodal benchmark functions from [21] is used to evaluate the performance of DSA. The description of the thirteen functions $f_1, \ldots, f_{13}$ can be found in Table I in [21]. The dimension of all functions is 30. Five functions f_1, f_6, f_7, f_8, f_9 are separable functions. The other eight are not separable. The performance of DSA is compared with eleven other classical EAs. The algorithms under this comparison are listed below. These EAs were chosen because they were evaluated in the above-mentioned test function suite in the references.

1. Differential Evolution (DE) [22]
2. Covariance matrix adaptation evolution strategy (CMA-ES) [23]
3. Genetic algorithm (GA)
4. Particle swarm optimization (PSO) [10]
5. Group search optimizer (GSO) [24]
6. Fast evolutionary programming (FEP) [21]
7. Classical evolutionary programming (CEP)
8. Fast evolution strategies (FES) [25]
9. Classical evolution strategies(CES) [26]
10. Fitness and Diversity-based DE (FFDE) [27]
11. Multi-Role-based DE (MRDE) [28]

The experiments and parameter settings are summarized as follows. The population size is 150 and the algorithm is run 25 times for each function. The maximum number of function evaluations is listed in Table 1. The parameter settings of other algorithms refer to the literature. Note that for DSA, the maximum number of evaluations is set to the same as or less than the other algorithms for comparison purposes.

5.1.2. Comparative analysis

The performance of an algorithm is measured by the mean and standard deviation of the best fitness value finally found in 25 runs. Based on the mean and standard deviation, we conducted independent t-tests to compare DSA with other algorithms pairwise for each function. This statistical test compares the means of two independent groups to determine whether there is a significant difference between them. Since the fitness values or error values that are very close to zero (such as 1.0e−8 and 1.0e−98) are not important in practice, but they affect the t-test. To address this issue, we follow a rule used in CEC competitions. If the fitness function value or error value is below 1.0e−8, it is adjusted to 1.0e−8. This adjustment is done before the t-test.

The experimental results are reported in Table 2. The results of other algorithms are taken from relevant literature. Table 2 gives the comparison of DSA with each algorithm. The symbol "+" indicates that DSA outperforms the comparison algorithm, "−" indicates that DSA performs worse than the comparison algorithm, and "=" indicates that the performance between DSA and the comparison algorithm is comparable.

Table 2 shows that DSA outperforms other EAs on most benchmark functions. However, an interesting phenomenon is observed from the results, that is, DSA performs worse than other EAs on functions f_5, f_8, f_{12}, f_{13}. The results show that DSA may perform well on most test functions, but may not be suitable for handling certain specific test functions.

5.1.3. Sensitivity analysis

We conduct sensitivity analysis on the number of particles in multi-point search direction. Principal Component Analysis (PCA) is used to determine this multi-point search direction, with the number of particles (points) being a critical parameter. In our computer simulation, 30 particles are randomly selected by default for multi-point direction search. To investigate how the number of randomly selected particles affects the performance of DSA, we conduct sensitivity analysis and comparative experiments with 10, 20, 30, 50 and 100 particles.

Table 3 show the influence of different numbers of particles on the algorithm. From the table, it is found that when the number of selected particles equals to 100, DSA has the best overall result. However, the data also show that increasing the number of particles does not necessarily lead to improved experimental results. For example, when the number of particles is 50, the experimental result is not better than when the number of particles is less than 50. When selecting fewer particles, it does not reflect the distribution of the entire population well; when selecting more particles, it will increase the computational complexity. Therefore, a trade-off value 30 is chosen in the experiment.

5.2. Evaluation on ISDA 2009 test functions and comparison with classical EAs

5.2.1. Experiment setting

We also evaluates the performance of DSA on high-dimensional benchmark functions with dimensions D = 50, 100, and 200, respectively. A set of thirteen benchmark functions is used in computer simulation. These functions comes from ISDA 2009 Workshop on Evolutionary Algorithms and other Meta heuristics for Continuous Optimization Problems - A Scalability Test.[2] Functions $f_1 - f_6$ are also the first

[2] https://sci2s.ugr.es/EAMHCO. Accessed on 1 June 2022.

Table 2
Results on the classical test functions and comparison with the classic evolutionary algorithms.

	Functions	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	+/ = /−
DE	Mean	2.66e−07	1.18e−03	1.34e+03	2.52e−01	2.11e+01	1.00e−08	1.38e−02	1.00e−08	1.73e+02	1.83e−04	2.87e−06	5.42e−07	1.00e−08	
	Std.	0.00e+00	3.87e−04	4.37e+02	2.45e−02	3.76e−01	0.00e+00	4.96e−03	1.15e+02	3.98e+00	3.00e−05	3.00e−06	0.00e+00	0.00e+00	
	Contest	+	+	+	+	−	=	+	=	+	+	+	−	−	8/2/3
CMA-ES	Mean	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	3.68e−03	1.00e−08	9.50e+00	1.00e−08	1.00e−08	1.00e−08	1.00e−08	
	Std.	7.59e−60	4.42e−29	7.90e−101	1.87e−22	5.32e−12	0.00e+00	9.22e−04	7.62e+02	5.67e+00	0.00e+00	0.00e+00	6.04e−32	7.30e−31	
	Contest	=	=	=	=	−	=	+	=	+	=	=	−	−	2/8/3
GA	Mean	3.17e+00	5.77e−01	9.75e+03	7.96e+00	3.39e+02	3.70e+00	1.05e−01	1.26e+04	6.51e−01	8.68e−01	1.00e+00	4.36e−02	1.68e−01	
	Std.	1.66e+00	1.31e−01	2.59e+03	1.51e+00	3.61e+02	1.95e+00	3.62e−02	2.11e+00	3.60e−01	2.81e−01	6.75e−02	5.06e−02	7.07e−02	
	Contest	+	+	+	+	+	+	+	+	+	+	+	−	+	12/0/1
PSO	Mean	1.00e−08	1.00e−08	1.20e−03	4.12e−01	3.74e+01	1.46e−01	9.90e−03	1.00e−08	2.08e+01	1.34e−03	2.32e−01	3.95e−02	5.05e−02	
	Std.	2.46e−36	1.14e−23	2.11e−03	2.50e−01	3.21e+01	4.18e−01	3.54e−02	4.64e+02	5.94e+00	4.24e−02	4.43e−01	9.14e−02	5.69e−01	
	Contest	=	=	+	+	=	=	=	=	+	=	+	−	=	4/8/1
GSO	Mean	1.95e−08	3.70e−05	5.78e+00	1.08e−01	4.98e+01	1.60e−02	7.38e−02	1.00e−08	1.02e+00	2.65e−05	3.08e−02	1.00e−08	4.69e−05	
	Std.	1.16e−08	8.62e−05	3.68e+00	4.00e−02	3.02e+01	1.33e−01	9.26e−02	2.21e−02	9.51e−01	3.08e−05	3.09e−02	9.17e−11	7.00e−04	
	Contest	+	+	+	+	+	=	+	=	+	+	+	−	=	9/3/1
FEP	Mean	5.70e−04	8.10e−03	1.60e−02	3.00e−01	5.06e+00	1.00e−08	7.60e−03	1.00e−08	4.60e−02	1.80e−02	1.60e−02	9.20e−06	1.60e−04	
	Std.	1.30e−04	7.70e−04	1.40e−02	5.00e−01	5.87e+00	0.00e+00	2.60e−03	5.26e+01	1.20e−02	1.20e−02	2.20e−02	6.14e−05	7.30e−05	
	Contest	+	+	+	+	−	=	+	=	+	+	+	−	=	8/3/2
CEP	Mean	2.20e−04	2.60e−03	5.00e−02	2.00e+00	6.17e+00	5.77e+02	1.80e−02	1.00e−08	8.90e+01	9.20e+00	8.60e−02	1.76e+00	1.40e+00	
	Std.	5.90e−04	1.70e−04	6.60e−02	1.20e+00	1.36e+01	1.13e+03	6.40e−03	6.34e+02	2.31e+01	2.80e+00	1.20e−01	2.40e+00	3.70e+00	
	Contest	=	+	+	+	−	+	+	=	+	+	+	=	=	8/4/1
FES	Mean	2.50e−04	6.00e−02	1.40e−03	5.50e−03	3.33e+01	1.00e−08	1.20e−02	1.00e−08	1.60e−01	1.20e−02	3.70e−02	2.80e−02	4.70e−05	
	Std.	6.80e−04	9.60e−03	5.30e−04	6.50e−04	4.31e+01	0.00e+00	5.80e−03	3.25e+01	3.30e−01	1.80e−03	5.00e−02	8.10e−11	1.50e−05	
	Contest	=	+	+	+	=	=	+	=	+	+	+	−	−	7/4/2
CES	Mean	3.40e−05	2.10e−02	1.30e−04	3.50e−01	6.69e+00	4.11e+02	3.00e−02	1.00e−08	7.08e+01	9.07e+00	3.80e−01	1.18e+00	1.39e+00	
	Std.	8.60e−06	2.20e−03	8.50e−05	4.20e−01	1.45e+01	6.59e+02	1.50e−02	6.31e+02	2.15e+01	2.84e+00	7.70e−01	1.87e+00	3.33e+00	
	Contest	+	+	+	+	−	+	+	=	+	+	+	=	+	10/2/1
FDDE	Mean	1.00e−08	4.70e−05	1.04e+03	2.10e−05	1.10e+01	1.00e−08	8.60e−03	1.00e−08	1.83e+02	1.00e−08	1.00e−08	1.00e−08	1.00e−08	
	Std.	0.00e+00	0.00e+00	3.06e+02	7.00e−06	5.97e−01	0.00e+00	2.51e−03	4.14e+01	5.69e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	
	Contest	=	+	+	+	−	=	+	=	+	=	=	−	−	5/5/3
MRDE	Mean	1.71e−07	3.02e−03	3.02e+04	4.49e−02	4.07e+01	1.00e−08	2.62e−02	1.00e−08	4.68e+01	4.30e−05	1.03e−07	2.29e−07	1.00e−08	
	Std.	0.00e+00	5.95e−04	7.83e+02	4.21e−03	1.27e+01	0.00e+00	7.44e−03	3.68e+00	3.82e+00	7.00e−06	0.00e+00	0.00e+00	0.00e+00	
	Contest	+	+	+	+	+	=	+	=	+	+	+	−	−	9/2/2
DSA	Mean	1.00e−08	1.00e−08	1.00e−08	1.00e−08	2.90e+01	1.00e−08	3.71e−05	1.00e−08	1.00e−08	1.00e−08	1.00e−08	8.08e−01	1.22e−04	
	Std	0.00e+00	0.00e+00	0.00e+00	0.00e+00	9.40e−03	0.00e+00	2.00e−05	1.24e+02	0.00e+00	0.00e+00	0.00e+00	7.66e−02	1.18e−04	

Table 3
A comparison of DSA with different number of particles in PCA.

		10	20	30	50	100
F1	Mean	**1.52E−297**	5.27E−292	5.80E−294	4.45E−292	1.19E−292
	Std.	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F2	Mean	7.34E−159	5.63E−167	1.83E−169	8.99E−169	**1.32E−173**
	Std.	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F3	Mean	4.17E−89	**5.76E−279**	1.82E−108	8.11E−203	1.30E−34
	Std.	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F4	Mean	1.67E−148	1.68E−153	4.12E−160	4.68E−168	**2.03E−174**
	Std.	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F5	Mean	2.90E+01	2.90E+01	2.90E+01	2.89E+01	**2.89E+01**
	Std.	2.84E−03	5.36E−02	9.37E−03	4.40E−02	3.32E−02
F6	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std.	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F7	Mean	6.77E−05	5.04E−05	**3.71**E−05	6.76E−05	1.20E−04
	Std.	7.00E−05	4.20E−05	2.00E−05	1.80E−05	8.70E−05
F8	Mean	−4.87E+03	−5.11E+03	−4.45E+03	−5.04E+03	−5.08E+03
	Std.	5.96E+02	3.97E+03	1.24E+02	9.01E+03	7.03E+03
F9	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std.	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F10	Mean	4.44E+16	4.44E+16	4.44E+16	4.44E+16	4.44E+16
	Std.	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F11	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std.	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F12	Mean	9.23E+01	8.95E+01	8.08E+01	9.18E+01	**7.63E+01**
	Std.	8.50E+02	7.71E+02	7.66E+02	8.62E+02	3.57E+02[2]
F13	Mean	6.90E−05	4.94E−05	1.22E−04	5.04E−05	**2.97E−05**
	Std.	3.30E−05	2.80E−05	1.18E−04	3.90E−05	3.40E−05

Table 4
Results on the 50D test functions and comparison with the classic evolutionary algorithms.

	Functions	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	+/ = /−
DE	Mean	1.00e−08	1.78e+01	3.61e+01	3.22e+02	1.00e−08	1.00e−08	1.00e−08	9.12e+01	2.69e−01	1.00e−08	3.14e−01	
	Std.	0.00e+00	5.72e+00	9.21e−01	2.03e+01	0.00e+00	5.20e−10	3.06e−09	3.70e+01	1.18e−02	0.00e+00	9.94e−02	
	Test	=	+	−	+	=	=	=	+	+	=	+	5/5/1
ODE	Mean	1.00e−08	1.00e−08	3.68e+01	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	2.70e−01	
	Std.	0.00e+00	0.00e+00	2.53e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	3.36e−02	
	Test	=	=	−	=	=	=	=	=	=	=	+	1/9/1
JADE	Mean	1.00e−08	1.00e−08	2.39e+00	3.92e+01	1.00e−08	1.00e−08	1.00e−08	1.00e−08	3.01e−03	6.30e−01	1.19e−02	
	Std.	0.00e+00	1.61e−13	2.18e+00	3.85e+01	0.00e+00	0.00e+00	0.00e+00	0.00e+00	3.33e−03	9.39e−01	1.51e−02	
	Test	=	=	−	=	=	=	=	=	=	=	=	0/10/1
SADE	Mean	1.00e−08	2.71e+00	2.99e+01	3.19e−06	1.00e−08	1.00e−08	1.00e−08	8.37e+02	2.26e−05	1.00e−08	3.62e−05	
	Std.	0.00e+00	8.30e−01	8.32e+00	7.08e−06	0.00e+00	0.00e+00	0.00e+00	1.57e+02	8.52e−06	0.00e+00	3.54e−05	
	Test	=	+	−	=	=	=	=	+	+	=	=	3/7/1
CME	Mean	8.86e−06	1.83e+01	4.39e+01	9.31e+01	8.42e−06	1.32e+01	1.01e−01	1.92e+01	2.99e+02	1.11e+01	3.20e+02	
	Std.	7.90e−07	5.64e+00	8.31e−01	1.32e+01	1.74e−07	8.72e−01	1.32e−01	8.56e+00	3.06e+01	2.95e+00	2.59e+01	
	Test	+	+	−	+	+	+	=	+	+	+	+	9/1/1
ME	Mean	7.68e−06	3.25e+01	6.99e+01	5.03e+01	2.96e−03	1.49e+01	2.86e−03	5.57e−02	3.11e+02	1.27e+00	2.82e+02	
	Std.	8.63e−07	5.33e+00	5.38e+01	6.43e+00	5.91e−03	6.64e−01	3.53e−03	3.24e−02	3.38e+01	1.03e+00	2.24e+01	
	Test	+	+	=	+	=	+	=	+	+	+	+	8/3/0
FDDE	Mean	1.00e−08	1.62e−01	5.95e+06	3.30e+02	1.48e−03	1.00e−08	1.00e−08	4.41e+01	2.65e+01	1.00e−08	2.80e−01	
	Std.	0.00e+00	1.04e−01	7.80e+05	1.98e+01	2.96e−03	0.00e+00	0.00e+00	8.24e+00	1.63e−01	0.00e+00	5.99e−02	
	Test	=	+	+	+	=	=	=	+	+	=	+	6/5/0
MRDE	Mean	3.51e−07	2.37e+00	4.60e+01	1.21e+02	1.60e−06	9.68e−05	1.05e−04	1.16e+04	1.09e+01	5.51e−07	1.07e+01	
	Std.	0.00e+00	1.89e−01	1.08e+00	9.16e+00	2.00e−06	1.10e−05	7.00e−06	1.42e+03	1.00e+00	0.00e+00	6.25e−01	
	Test	+	+	−	+	=	+	+	+	+	+	+	9/1/1
DSA	Mean	1.00e−08	1.00e−08	4.90e+01	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	
	Std.	0.00e+00	0.00e+00	2.89e−02	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	

Table 5
Results on the 100D test functions and comparison with the classic evolutionary algorithms.

	Functions	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	+/ = /−
DE	Mean	1.00e−08	5.21e+01	2.17e+02	6.38e+02	1.00e−08	1.00e−08	1.00e−08	3.94e+03	1.14e+00	5.25e+00	5.17e+00	
	Std.	0.00e+00	4.12e+00	1.15e+02	1.01e+02	0.00e+00	1.33e−10	1.23e−11	7.69e+02	9.30e−01	1.29e+00	3.32e+00	
	Test	=	+	=	+	=	=	=	+	+	+	+	6/5/0
ODE	Mean	1.00e−08	1.00e−08	1.43e+02	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	3.46e+00	
	Std.	0.00e+00	0.00e+00	8.22e+01	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	1.73e+00	
	Test	=	=	=	=	=	=	=	=	=	=	+	1/10/0
JADE	Mean	1.00e−08	6.52e−01	3.23e+00	4.25e+02	3.45e−03	9.65e−01	1.00e−08	5.17e−05	2.43e+00	1.78e+01	3.36e+00	
	Std.	0.00e+00	2.69e−01	1.80e+00	1.14e+02	4.80e−03	9.43e−01	0.00e+00	7.81e−05	2.42e+00	2.20e+00	3.75e+00	
	Test	=	+	−	+	=	=	=	=	=	+	=	3/7/1
SADE	Mean	1.00e−08	9.75e+00	1.29e+02	2.34e+01	1.00e−08	1.00e−08	1.00e−08	8.51e+03	2.15e−03	2.52e+00	1.00e−08	
	Std.	0.00e+00	6.75e−01	4.76e+01	5.33e+00	0.00e+00	0.00e+00	0.00e+00	9.34e+02	4.80e−03	5.75e−01	1.28e−09	
	Test	=	+	=	+	=	=	=	+	=	+	=	4/7/0
CME	Mean	1.22e−05	4.50e+01	1.10e+02	3.94e+02	9.60e−06	1.78e+01	9.74e−01	3.79e+02	7.90e+02	3.41e+01	7.72e+02	
	Std.	8.30e−06	3.49e+00	3.53e+01	2.47e+01	2.43e−07	2.95e−01	3.43e−01	1.12e+02	2.28e+01	1.40e+00	3.07e+01	
	Test	+	+	=	+	+	+	+	+	+	+	+	10/1/0
ME	Mean	5.48e−05	5.65e+01	1.22e+02	1.55e+02	9.74e−06	1.74e+01	2.96e+02	2.06e+01	6.87e+02	1.08e+01	7.25e+02	
	Std.	5.75e−05	3.21e+00	3.36e+01	1.92e+01	1.63e−07	3.55e−01	2.54e+01	5.05e+00	4.24e+01	2.50e+00	3.69e+01	
	Test	=	+	=	+	+	+	+	+	+	+	+	9/2/0
FDDE	Mean	1.00e−08	8.96e+00	1.15e+07	7.29e+02	1.97e−03	1.00e−08	1.00e−08	4.38e+03	1.89e+00	3.99e+00	3.92e+01	
	Std.	0.00e+00	1.61e+00	2.07e+05	5.33e+01	3.94e−03	0.00e+00	0.00e+00	1.41e+03	1.12e+00	2.60e+00	2.46e+00	
	Test	=	+	+	+	=	=	=	+	+	+	+	7/4/0
MRDE	Mean	4.21e−07	8.63e+00	1.21e+02	4.01e+02	2.73e−07	9.59e−05	2.49e−04	5.22e+04	3.31e+01	1.28e−06	3.53e+01	
	Std.	0.00e+00	8.22e−01	2.94e+01	1.34e+01	0.00e+00	4.00e−06	2.90e−05	3.50e+03	1.65e+00	0.00e+00	2.31e+00	
	Test	+	+	=	+	+	+	+	+	+	+	+	10/1/0
DSA	Mean	1.00e−08	1.00e−08	9.90e+01	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	
	Std.	0.00e+00	0.00e+00	1.43e−02	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	

six of seven benchmark functions used in the CEC 2008 Special Session and Competition on Large Scale Global Optimization.[3] Four functions are separable f_1, f_4, f_6, f_5, while the other seven are non-separable.

The performance of DSA is compared with eight other EAs in the case of high-dimensional benchmark functions. DE and memetic evolution were chosen because of their better performance on high-dimensional functions. The EAs included in this comparison are listed below.

1. Differential Evolution (DE) [22]
2. Opposition-based Differential Evolution (ODE) [29]
3. JADE [30]
4. Self-adaptive Differential Evolution (SADE) [31]
5. Cellular Memetic Evolution (CME) [32]

[3] https://github.com/P-N-Suganthan/CEC2008. Accessed on 1 June 2022.

Table 6
Results on the 200D test functions and comparison with the classic evolutionary algorithms.

	Functions	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	+/=/−
DE	Mean	1.00e−08	8.26e+01	4.14e+02	3.20e+02	3.91e−02	1.76e+00	1.00e−08	2.29e+04	8.44e+01	3.42e+01	9.03e+01	
	Std.	0.00e+00	4.68e+00	8.36e+01	1.98e+01	7.95e−02	5.02e−01	1.38e−09	2.84e+03	2.95e+01	3.91e+00	1.87e+01	
	Test	=	+	+	+	=	+	=	+	+	+	+	8/3/0
ODE	Mean	1.00e−08	5.23e+01	4.64e+02	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.21e+02	
	Std.	0.00e+00	5.25e+00	7.55e+01	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	3.08e+01	
	Test	=	+	+	=	=	=	=	=	=	=	+	3/8/0
JADE	Mean	1.00e−08	1.65e+01	1.65e+01	1.31e+03	1.00e−08	2.48e+00	1.00e−08	3.75e+01	1.01e+01	7.56e+01	8.18e+00	
	Std.	0.00e+00	1.09e+00	3.23e+01	7.70e+01	0.00e+00	3.09e−01	0.00e+00	4.72e+01	8.32e+00	2.70e+00	1.12e+01	
	Test	=	+	−	+	=	+	=	=	+	+	=	5/5/1
SADE	Mean	1.00e−08	1.56e+01	3.85e+02	1.24e+02	1.48e−03	1.70e+00	1.00e−08	3.54e+02	3.16e+00	4.36e+01	2.12e+00	
	Std.	0.00e+00	9.53e−01	4.84e+01	3.51e+00	3.31e−03	1.72e−01	0.00e+00	3.21e+03	4.01e+00	4.27e+00	2.62e+00	
	Test	=	+	+	+	=	+	=	=	=	+	=	5/6/0
CME	Mean	4.41e−03	6.96e+01	1.97e+02	1.06e+03	1.32e−03	1.91e+01	4.55e+00	7.81e+03	1.64e+03	7.10e+01	1.63e+03	
	Std.	5.47e−03	1.81e+00	4.39e+00	5.55e+01	4.73e−04	6.14e−02	1.64e+00	1.36e+03	1.94e+01	4.35e+00	1.96e+01	
	Test	=	+	=	+	+	+	+	+	+	+	+	9/2/0
ME	Mean	5.35e−03	7.27e+01	2.51e+02	5.59e+02	6.60e−05	1.83e+01	1.03e+00	2.12e+03	1.54e+03	5.27e+01	1.53e+03	
	Std.	1.99e−03	1.88e+00	4.53e+01	5.49e+01	1.45e−05	1.84e−01	1.60e+00	3.72e+02	4.39e+01	6.40e+00	2.54e+01	
	Test	+	+	+	+	+	+	=	+	+	+	+	10/1/0
FDDE	Mean	1.00e−08	2.72e+01	2.63e+07	1.52e+02	1.00e−08	1.43e+03	1.00e−08	2.93e+04	6.43e+01	3.54e+01	6.99e+01	
	Std.	0.00e+00	8.26e−01	1.94e+05	1.71e+01	0.00e+00	2.28e−01	0.00e+00	4.71e+04	1.02e+01	3.64e+00	1.34e+01	
	Test	=	+	+	+	=	+	=	=	+	+	+	7/4/0
MRDE	Mean	1.30e−06	2.92e+01	2.06e+02	1.11e+03	2.67e−07	9.77e−05	6.66e−04	2.01e+05	7.81e+01	2.79e−06	7.82e+01	
	Std.	0.00e+00	2.04e+00	2.28e+01	2.48e+01	0.00e+00	8.00e−06	4.10e−05	8.20e+03	1.73e+00	1.00e−06	1.87e+00	
	Test	+	+	=	+	+	+	+	+	+	+	+	10/1/0
DSA	Mean	1.00e−08	1.00e−08	1.99e+02	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	
	Std.	0.00e+00	0.00e+00	2.06e−01	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	

6. Memetic Evolution (ME) [33,34]
7. Fitness and Diversity-based DE (FFDE) [27]
8. Multi-Role-based DE (MRDE) [27,28]

Five independent runs were completed on each test function. The population is set to 150. The maximum number of fitness evaluations is $5000D$, where D is the dimension of a function. Each run stops when the maximum number of evaluations has been achieved.

5.2.2. Comparative analysis

The performance of EAs is evaluated in terms of mean and standard deviation of errors. This is different from Section 5.1 which is compares fitness values. Tables 4, 5, and 6, report experimental results for $D = 50, D = 100$, and $D = 200$. Any result with its value below 1E–14 has been treated as 0.0. Using the mean and standard deviation, we performed independent t-tests to compare DSA with other algorithms in pairs. The symbols "+", "−", and "=" indicate that the performance of DSA is better than, worse than, or similar to the comparison algorithm, respectively.

Tables 4, 5, and 6 present the comparison of DSA with each algorithm. The comparisons show that DSA has achieved better performance than other EAs on the majority of benchmark functions across all dimensions. An exception is function f_3, where JADE achieved better mean results than DSA, but its variance results were worse. Experimental results confirm that DSA outperforms the other algorithms in high-dimensional benchmark functions.

5.3. Evaluation on CEC 2022 competition test functions and comparison with state of the art PSO algorithms

5.3.1. Experimental setting

To evaluate the performance of the proposed algorithm, we compared DSA with several recent PSO variants on the CEC 2022 competition benchmark. The following algorithms were used in the comparison.

1. Adaptive multistrategy ensemble particle swarm optimization (AMSEPSO) [35]
2. Modified particle swarm optimization using adaptive strategy (MPSO) [36]
3. Fitness-distance balance phasor particle swarm optimization (FDBPSO) [37]
4. Pyramid particle swarm optimization (PPSO) [38]
5. PSO on single-objective numerical optimization (PSOsono) [39]
6. Velocity pausing particle swarm optimization (VPPSO) [40]
7. Multi-strategy Particle Swarm Optimization with Adaptive Forgetting (AFMPSO) [41]

The parameters required for each algorithm are as shown in Table 1 in [41]. The dimension was set to 10, and the maximum number of evaluations was 200,000. Each algorithm was run 30 times independently, resulting in 30 samples.

5.3.2. Comparative analysis

Table 7 gives the experimental results and comparisons of each algorithm in 10 dimensions. It includes five key metrics: best, worst, median, mean, and standard deviation. Based on the mean, and standard deviation, we conducted an independent t-test to compare DSA with other algorithms pairwise for each function. The symbols "+", "−", and "=" indicate that the performance of DSA is better than, worse than, or similar to the comparison algorithm, respectively.

As shown in Table 7, DSA outperforms the 7 PSO variants overall in the 10-dimensional problem. Specifically, DSA outperforms or is equal to the FDBPSO, MPSO, and VPPSO algorithms, and is not inferior in any case. DSA outperforms or is equal to the PPSO, PSOsono, AFMPSO, and AMSEPSO algorithms in most problems, and performs worse in only one or two problems.

5.4. Evaluation on CEC 2022 competition benchmark and comparison with leading algorithms

5.4.1. Experimental setting

To evaluate the performance of the proposed algorithm, we further compare DSA with several leading algorithms in the CEC 2022 competition. The compared algorithms are as follows.

Table 7
Experimental comparison with seven state-of-art PSO variants.

F	Index	AMSEPSO	FDBPSO	MPSO	PPSO	PSOsono	VPPSO	AFMPSO	DSA
	Best	1.00e−08	1.86e+03	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Worst	1.00e−08	1.47e+04	1.91e−05	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
F1	Median	1.00e−08	6.81e+03	8.35e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Mean	1.00e−08	7.07e+03	1.63e−06	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Std	0.00E+00	3.16E+03	4.07E−06	0.00E+00	0.00E+00	8.65E−10	0.00E+00	0.00E+00
	Test	=	+	+	=	=	=	=	
	Best	1.99e−02	6.10e+00	5.72e−06	9.98e−04	1.00e−08	5.07e−04	1.00e−08	9.60e−05
	Worst	7.71e+00	8.78e+01	8.92e+00	8.92e+00	9.84e+00	7.08e+01	1.00e−08	8.92e+00
F2	Median	5.41e+00	3.42e+01	5.59e+00	4.92e+00	3.99e+00	4.10e+00	1.00e−08	2.64e−04
	Mean	4.42e+00	3.73e+01	4.43e+00	4.70e+00	5.22e+00	6.64e+00	1.00e−08	6.96e−01
	Std	2.47E+00	2.19E+01	4.04E+00	3.93E+00	3.19E+00	1.27E+01	0.00E+00	1.94E+00
	Test	+	+	+	+	+	+	=	
	Best	1.00e−08	6.15e+00	2.01e−07	1.00e−08	1.00e−08	8.41e−05	3.18e−02	1.00e−08
	Worst	1.00e−08	3.16e+01	2.30e−04	1.13e−05	5.21e−02	8.48e+00	7.70e−01	1.00e−08
F3	Median	1.00e−08	1.47e+01	3.48e−06	1.00e−08	1.10e−05	6.87e−01	3.21e−01	1.00e−08
	Mean	1.00e−08	1.55e+01	1.61e−05	5.86e−07	5.86e−03	1.09e+00	3.41e−01	1.00e−08
	Std	0.00E+00	6.37E+00	4.22E−05	2.15E−06	1.26E−02	1.75E+00	2.29E−01	2.42E−11
	Test	=	+	+	=	+	+	+	
	Best	1.00e−08	8.82e+00	2.98e+00	1.00e−08	1.99e+00	9.95e+00	3.98e+00	9.95e−01
	Worst	9.09e+00	4.06e+01	1.89e+01	5.97e+00	1.69e+01	3.48e+01	3.18e+01	2.39e+01
F4	Median	2.33e+00	2.10e+01	8.46e+00	2.98e+00	7.96e+00	1.89e+01	1.24e+01	8.95e+00
	Mean	3.09e+00	2.07e+01	9.15e+00	2.82e+00	7.99e+00	1.86e+01	1.33e+01	9.65e+00
	Std	2.52E+00	6.96E+00	4.13E+00	1.43E+00	3.58E+00	6.26E+00	6.86E+00	5.01E+00
	Test	−	+	=	−	=	+	+	
	Best	1.00e−08	8.56e+00	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Worst	1.00e−08	8.08e+02	4.54e−01	4.54e−01	8.95e−02	3.31e+00	1.00e−08	1.00e−08
F5	Median	1.00e−08	2.09e+02	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Mean	1.00e−08	2.65e+02	5.44e−02	1.81e−02	5.97e−03	2.32e−01	1.00e−08	1.00e−08
	Std	0.00E+00	1.83E+02	1.38E−01	8.40E−02	2.27E−02	6.38E−01	0.00E+00	3.81E−13
	Test	=	+	+	=	=	=	=	
	Best	5.83e+00	2.61e+04	4.29e+00	7.84e+00	2.35e+00	1.55e+02	1.03e+01	1.53e+00
	Worst	3.38e+02	4.31e+05	3.51e+03	3.07e+03	1.21e+03	6.22e+03	2.20e+02	2.44e+03
F6	Median	6.43e+01	8.83e+04	2.45e+02	2.83e+02	3.73e+01	1.80e+03	4.54e+01	4.38e+02
	Mean	8.28e+01	1.32e+05	6.04e+02	6.21e+02	1.25e+02	2.37e+03	5.23e+01	6.50e+02
	Std	7.40E+01	1.09E+05	8.30E+02	7.55E+02	2.46E+02	2.10E+03	4.10E+01	6.44E+02
	Test	−	+	=	=	−	+	−	
	Best	1.70e−06	2.61e+01	3.48e−04	4.68e−03	8.10e−08	1.06e+00	8.33e+00	1.00e−08
	Worst	7.57e+00	7.65e+01	2.40e+01	2.54e+01	2.49e+01	4.49e+01	3.91e+01	2.00e+01
F7	Median	8.50e−01	5.60e+01	1.43e+01	7.29e+00	1.28e+01	2.49e+01	2.45e+01	1.60e−06
	Mean	2.27e+00	5.61e+01	1.26e+01	1.21e+01	1.23e+01	2.60e+01	2.35e+01	1.96e+00
	Std	2.41E+00	1.28E+01	9.59E+00	1.00E+01	1.02E+01	9.92E+00	6.76E+00	5.07E+00
	Test	=	+	+	+	+	+	+	
	Best	1.13e+00	2.11e+01	2.41e−01	4.10e−01	2.39e−01	1.63e+00	2.96e+00	1.65e−04
	Worst	2.37e+01	4.58e+01	2.61e+01	2.27e+01	2.26e+01	2.64e+01	2.64e+01	2.02e+01
F8	Median	7.56e+00	3.18e+01	2.19e+01	2.08e+01	2.02e+01	2.22e+01	2.24e+01	6.33e−01
	Mean	9.48e+00	3.22e+01	1.84e+01	1.99e+01	1.32e+01	1.97e+01	2.07e+01	4.26e+00
	Std	6.94E+00	5.09E+00	8.67E+00	4.00E+00	9.39E+00	7.45E+00	5.98E+00	7.92E+00
	Test	+	+	+	+	+	+	+	
	Best	2.29e+02	2.36e+02	2.29e+02	2.29e+02	2.30e+02	2.29e+02	2.29e+02	2.29e+02
	Worst	2.31e+02	4.19e+02	2.29e+02	2.29e+02	2.33e+02	2.29e+02	2.34e+02	2.29e+02
F9	Median	2.30e+02	3.57e+02	2.29e+02	2.29e+02	2.31e+02	2.29e+02	2.30e+02	2.29e+02
	Mean	2.30e+02	3.45e+02	2.29e+02	2.29e+02	2.31e+02	2.29e+02	2.31e+02	2.29e+02
	Std	2.61E−01	5.11E+01	0.00E+00	0.00E+00	6.02E−01	2.77E−09	1.26E+00	6.27E−14
	Test	+	+	=	=	+	=	+	
	Best	1.00e+02	1.01e+02	1.00e+02	1.00e+02	1.00e+02	1.00e+02	1.00e+02	1.00e+02
	Worst	1.00e+02	2.67e+02	2.15e+02	2.12e+02	2.17e+02	2.19e+02	1.01e+02	1.00e+02

(*continued on next page*)

1. Evolutionary Algorithm with Eigen crossover (EA4eig) [42]. The 1st-ranked algorithm.
2. Non-Linear population size reduction Success-History Adaptive Differential Evolution with Linear Bias Change (NL-SHADE-LBC) [43]. The 2nd-ranked algorithm.
3. An enhanced version of NLSHADE-RSP with Midpoint (NLSHADE-RSP-MID) [44]. The 3rd-ranked algorithm.
4. An adaptive variant of differential evolution combining binomial and exponential crossover with feature transformation (jSObeE) [45]. The 5th-ranked algorithm.
5. SHADE with tolerance-based multiple topology selection framework (MTT_SHADE) [46]. The 6th-ranked algorithm.

The dimension was set to 10, and the maximum number of evaluations was 200,000. The results includes five key metrics: best, worst,

Table 7 (*continued*).

F	Index	AMSEPSO	FDBPSO	MPSO	PPSO	PSOsono	VPPSO	AFMPSO	DSA
F10	Median	1.00e+02	1.02e+02	1.00e+02	1.01e+02	1.00e+02	1.00e+02	1.00e+02	1.00e+02
	Mean	1.00e+02	1.30e+02	1.33e+02	1.42e+02	1.19e+02	1.04e+02	1.00e+02	1.00e+02
	Std	2.93E−02	5.82E+01	5.03E+01	5.23E+01	4.17E+01	2.17E+01	8.81E−02	2.69E−03
	Test	=	+	+	+	+	=	=	
	Best	1.00e−08	1.30e+02	2.71e−05	1.00e−08	1.00e−08	3.19e−04	1.00e−08	1.00e−08
	Worst	1.54e+02	3.19e+02	3.00e+02	3.00e+02	4.00e+02	3.00e+02	1.00e−08	1.00e−08
F11	Median	1.00e−08	1.75e+02	6.79e−03	1.50e+02	1.00e−08	5.03e−04	1.00e−08	1.00e−08
	Mean	1.52e+01	1.94e+02	6.04e+01	1.30e+02	8.68e+01	4.31e+01	1.00e−08	1.00e−08
	Std	4.23E+01	4.43E+01	1.08E+02	1.10E+02	1.19E+02	1.03E+02	0.00E+00	0.00E+00
	Test	=	+	+	+	+	+	=	
	Best	1.62e+02	1.65e+02	1.60e+02	1.63e+02	1.65e+02	1.59e+02	1.63e+02	1.59e+02
	Worst	1.65e+02	2.40e+02	1.81e+02	1.70e+02	1.67e+02	1.65e+02	1.69e+02	1.67e+02
F12	Median	1.65e+02	1.72e+02	1.64e+02	1.65e+02	1.65e+02	1.61e+02	1.66e+02	1.63e+02
	Mean	1.65e+02	1.79e+02	1.65e+02	1.66e+02	1.65e+02	1.61e+02	1.66e+02	1.62e+02
	Std	4.83E−01	1.77E+01	3.80E+00	1.49E+00	4.97E−01	1.66E+00	1.63E+00	2.74E+00
	Test	+	+	+	+	+	=	+	
	+/ = /−	4/6/2	12/0/0	9/3/0	6/5/1	8/3/1	7/5/0	6/5/1	

Table 8
Experimental comparison with leading EAs in the CEC 2022 competition.

F	Index	EA4eig	NL-SHADE-LBC	NLSHADE-RSP-MID	jSObeE	MTT_SHADE	DSA
	Best	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Worst	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
F1	Median	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Mean	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Std	0.00E+00	0.00e+00	0.00e+00	1.77e−09	0.00e+00	0.00E+00
	Test	=	=	=	=	=	
	Best	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	9.60e−05
	Worst	3.99e+00	1.00e−08	1.00e−08	8.92e+00	8.92e+00	8.92e+00
F2	Median	1.00e−08	1.00e−08	1.00e−08	3.99e+00	3.99e+00	2.64e−04
	Mean	1.46e+00	1.33e−01	1.00e−08	5.17e+00	5.00e+00	6.96e−01
	Std	1.95E+00	7.16e−01	0.00e+00	2.41e+00	2.31e+00	1.94E+00
	Test	=	=	=	+	+	
	Best	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Worst	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
F3	Median	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Mean	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Std	0.00E+00	0.00e+00	0.00e+00	1.04e−09	0.00e+00	2.42E−11
	Test	=	=	=	=	=	
	Best	1.00e−08	2.16e−05	2.98e+00	1.99e+00	1.99e+00	9.95e−01
	Worst	3.98e+00	2.99e+00	2.09e+01	4.97e+00	8.95e+00	2.39e+01
F4	Median	9.95e−01	9.95e−01	9.45e+00	2.98e+00	3.98e+00	8.95e+00

(*continued on next page*)

median, mean, and standard deviation. Each algorithm was run 30 times independently, resulting in 30 samples.

5.4.2. Comparative analysis

Table 8 presents the experimental and comparative results of each algorithm across 10 dimensions, including five key indicators: best, worst, median, mean, and standard deviation. Using the mean and standard deviation, we conducted independent t-tests to compare DSA with other algorithms on a pairwise basis for each function. The symbols "+", "−", and "=" indicate that the performance of DSA is better than, worse than, or similar to the comparison algorithm, respectively.

Table 8 shows that DSA performs worse than EA4eig, the winning algorithm of the CEC 2022 competition. Nevertheless, its performance is comparable to other participating algorithms in most benchmark functions. It is worth noting that EA4eig is an ensemble of four adaptive evolutionary algorithms (CMA-ES, CoBiDE, an adaptive variant of jSO, and IDE). This significantly improves its performance and consolidates its leading position. It is worth noting that all top ten algorithms are variants of DE. One advantage of DE over PSO is the use of crossover. While DSA does not show an advantage over these leading DE variants, it remains competitive and shows consistently strong performance. Similar observations were reported in [41] where PSO was compared with the best performing EA algorithm in the CEC 2022 competition.

6. Conclusion

This paper proposes a new evolutionary algorithm, called the multiple direction search algorithm (DSA). DSA combines different search directions in a weighted sum manner. Theoretical analysis shows that under mild conditions, the rate of convergence along the weighted direction is no worse than the rate of convergence along the best search direction by a positive constant, and may exceed it in some cases.

To verify the above theoretical results, we designed and implemented a simple DSA algorithm and tested it through computer simulation. DSA integrates five search directions, including a multi-point search direction designed by principal component analysis. Experimental results on the selected benchmark test sets show that DSA outperforms classical eight evolutionary algorithms and seven state-of-the-art particle swarm optimization algorithms in most benchmarks. Furthermore, the experimental results also show that DSA, like other PSO algorithms, does not outperform the leading algorithms participating in the CEC 2022 competition, because these DE-based algorithms utilize crossover, while PSO does not.

Table 8 (*continued*).

F	Index	EA4eig	NL-SHADE-LBC	NLSHADE-RSP-MID	jSObeE	MTT_SHADE	DSA
	Mean	1.29e+00	1.30e+00	1.00e+01	3.22e+00	4.01e+00	9.65e+00
	Std	1.05E+00	7.78e−01	4.55e+00	8.13e−01	1.56e+00	5.01E+00
	Test	−	−	=	−	−	
F5	Best	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Worst	1.00e−08	1.00e−08	1.66e+01	1.00e−08	1.00e−08	1.00e−08
	Median	1.00e−08	1.00e−08	3.17e−01	1.00e−08	1.00e−08	1.00e−08
	Mean	1.00e−08	1.00e−08	1.69e+00	1.00e−08	1.00e−08	1.00e−08
	Std	0.00E+00	0.00e+00	3.88e+00	1.47e−09	0.00e+00	3.81E−13
	Test	=	=	+	=	=	
F6	Best	2.71e−04	4.31e−03	1.78e−02	1.60e−03	2.56e−02	1.53e+00
	Worst	2.51e−01	4.40e−01	1.04e+00	3.58e−01	5.00e−01	2.44e+03
	Median	5.66e−03	7.21e−02	9.65e−02	1.45e−02	3.22e−01	4.38e+02
	Mean	2.42e−02	1.24e−01	1.67e−01	4.36e−02	3.10e−01	6.50e+02
	Std	5.58E−02	1.25e−01	2.45e−01	7.30e−02	1.42e−01	6.44E+02
	Test	−	−	−	−	−	
F7	Best	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Worst	1.00e−08	1.00e−08	1.00e−08	6.38e−06	3.91e−01	2.00e+01
	Median	1.00e−08	1.00e−08	1.00e−08	1.12e−08	7.11e−02	1.60e−06
	Mean	1.00e−08	1.00e−08	1.00e−08	3.50e−07	8.47e−02	1.96e+00
	Std	0.00E+00	0.00e+00	0.00e+00	1.19e−06	8.56e−02	5.07E+00
	Test	−	−	−	−	−	
F8	Best	5.49e−04	1.00e−08	9.39e−05	3.54e−02	2.48e−02	1.65e−04
	Worst	3.05e−01	1.79e−01	6.58e−01	3.51e−01	2.03e+01	2.02e+01
	Median	1.21e−01	3.29e−02	8.09e−02	1.12e−01	2.91e+00	6.33e−01
	Mean	1.15e−01	4.60e−02	2.38e−01	1.31e−01	6.43e+00	4.26e+00
	Std	9.88E−02	3.80e−02	2.78e−01	7.94e−02	7.02e+00	7.92E+00
	Test	−	−	−	−	=	
F9	Best	1.86e+02	2.29e+02	2.29e+02	2.29e+02	2.29e+02	2.29e+02
	Worst	1.86e+02	2.29e+02	2.29e+02	2.29e+02	2.29e+02	2.29e+02
	Median	1.86e+02	2.29e+02	2.29e+02	2.29e+02	2.29e+02	2.29e+02
	Mean	1.86e+02	2.29e+02	2.29e+02	2.29e+02	2.29e+02	2.29e+02
	Std	0.00E+00	5.68e−14	0.00e+00	8.67e−14	0.00e+00	6.27E−14
	Test	−	=	=	=	=	
F10	Best	1.00e+02	1.00e+02	1.00e−08	1.00e+02	1.00e+02	1.00e+02
	Worst	1.00e+02	1.00e+02	1.00e+02	1.00e+02	2.05e+02	1.00e+02
	Median	1.00e+02	1.00e+02	1.00e−08	1.00e+02	1.00e+02	1.00e+02
	Mean	1.00e+02	1.00e+02	4.53e+00	1.00e+02	1.04e+02	1.00e+02
	Std	3.57E−02	2.95e−02	1.83e+01	2.39e−02	1.91e+01	2.69E−03
	Test	=	=	−	=	=	
F11	Best	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Worst	1.00e−08	1.00e−08	1.36e−08	1.00e−08	1.00e−08	1.00e−08
	Median	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08	1.00e−08
	Mean	1.00e−08	1.00e−08	1.01e−08	1.00e−08	1.00e−08	1.00e−08
	Std	0.00E+00	0.00e+00	6.50e−10	7.83e−10	0.00e+00	0.00E+00
	Test	=	=	=	=	=	
F12	Best	1.45e+02	1.63e+02	1.63e+02	1.62e+02	1.59e+02	1.59e+02
	Worst	1.59e+02	1.65e+02	1.66e+02	1.65e+02	1.65e+02	1.67e+02
	Median	1.46e+02	1.65e+02	1.65e+02	1.63e+02	1.62e+02	1.63e+02
	Mean	1.48e+02	1.65e+02	1.65e+02	1.63e+02	1.62e+02	1.62e+02
	Std	4.83E+00	4.04e−01	9.72e−01	9.18e−01	1.66e+00	2.74E+00
	Test	−	+	+	=	=	
	+/ = /−	0/6/6	1/7/4	2/6/4	1/7/4	1/8/3	

The current implementation of DSA is based on a simple weighted sum of search directions. One improvement for future research is to adjust the weights. In addition, DSA will also design and use other search directions.

CRediT authorship contribution statement

Wei Huang: Writing – original draft, Visualization, Methodology, Data curation, Conceptualization. **Jun He:** Writing – review & editing, Methodology, Formal analysis, Conceptualization. **Liehuang Zhu:** Resources, Methodology, Investigation, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Wei Huang reports financial support was provided by National Natural Science Foundation of China. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] C.L. Camacho-Villalón, M. Dorigo, T. Stützle, PSO-X: A component-based framework for the automatic design of particle swarm optimization algorithms, IEEE Trans. Evol. Comput. 26 (3) (2021) 402–416.
[2] X. Ji, Y. Zhang, D. Gong, X. Sun, Dual-surrogate-assisted cooperative particle swarm optimization for expensive multimodal problems, IEEE Trans. Evol. Comput. 25 (4) (2021) 794–808.
[3] L. Li, L. Chang, T. Gu, W. Sheng, W. Wang, On the norm of dominant difference for many-objective particle swarm optimization, IEEE Trans. Cybern. 51 (4) (2019) 2055–2067.
[4] X. Yu, W.-N. Chen, T. Gu, H. Zhang, H. Yuan, S. Kwong, J. Zhang, Set-based discrete particle swarm optimization based on decomposition for permutation-based multiobjective combinatorial optimization problems, IEEE Trans. Cybern. 48 (7) (2017) 2139–2153.
[5] Z.-J. Wang, Z.-H. Zhan, S. Kwong, H. Jin, J. Zhang, Adaptive granularity learning distributed particle swarm optimization for large-scale optimization, IEEE Trans. Cybern. 51 (3) (2021) 1175–1188, http://dx.doi.org/10.1109/TCYB.2020.2977956.
[6] J. Tian, Y. Tan, J. Zeng, C. Sun, Y. Jin, Multiobjective infill criterion driven gaussian process-assisted particle swarm optimization of high-dimensional expensive problems, IEEE Trans. Evol. Comput. 23 (3) (2018) 459–472.
[7] Y. Zhang, X.-F. Ji, X.-Z. Gao, D.-W. Gong, X.-Y. Sun, Objective-constraint mutual-guided surrogate-based particle swarm optimization for expensive constrained multimodal problems, IEEE Trans. Evol. Comput. 27 (4) (2022) 908–922.
[8] M.R. Bonyadi, Z. Michalewicz, Particle swarm optimization for single objective continuous space problems: a review, Evol. Comput. 25 (1) (2017) 1–54.
[9] Y. Chen, J. He, Average convergence rate of evolutionary algorithms in continuous optimization, Inform. Sci. 562 (2021) 200–219.
[10] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95-International Conference on Neural Networks, Vol. 4, IEEE, 1995, pp. 1942–1948.
[11] Y. Shi, R. Eberhart, A modified particle swarm optimizer, in: 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360), IEEE, 1998, pp. 69–73.
[12] M. Clerc, J. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, IEEE Trans. Evol. Comput. 6 (1) (2002) 58–73.
[13] M.R. Bonyadi, Z. Michalewicz, Stability analysis of the particle swarm optimization without stagnation assumption, IEEE Trans. Evol. Comput. 20 (5) (2015) 814–819.
[14] M.R. Bonyadi, Z. Michalewicz, A locally convergent rotationally invariant particle swarm optimization algorithm, Swarm Intell. 8 (3) (2014) 159–198.
[15] H. Zhang, M. Yuan, Y. Liang, Q. Liao, A novel particle swarm optimization based on prey–predator relationship, Appl. Soft Comput. 68 (2018) 202–218.
[16] C. Sun, Y. Jin, R. Cheng, J. Ding, J. Zeng, Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems, IEEE Trans. Evol. Comput. 21 (4) (2017) 644–660.
[17] S. Helwig, J. Branke, S. Mostaghim, Experimental analysis of bound handling techniques in particle swarm optimization, IEEE Trans. Evol. Comput. 17 (2) (2012) 259–271.
[18] C.W. Cleghorn, A.P. Engelbrecht, Particle swarm stability: a theoretical extension using the non-stagnate distribution assumption, Swarm Intell. 12 (1) (2018) 1–22.
[19] N. Hansen, R. Ros, N. Mauny, M. Schoenauer, A. Auger, Impacts of invariance in search: When CMA-ES and PSO face ill-conditioned and non-separable problems, Appl. Soft Comput. 11 (8) (2011) 5755–5769.
[20] W. Huang, T. Xu, K. Li, J. He, Multiobjective differential evolution enhanced with principle component analysis for constrained optimization, Swarm Evol. Comput. 50 (2019) 100571.
[21] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Trans. Evol. Comput. 3 (2) (1999) 82–102.
[22] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[23] N. Hansen, S.D. Müller, P. Koumoutsakos, Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES), Evol. Comput. 11 (1) (2003) 1–18.
[24] S. He, Q.H. Wu, J.R. Saunders, Group search optimizer: an optimization algorithm inspired by animal searching behavior, IEEE Trans. Evol. Comput. 13 (5) (2009) 973–990.
[25] X. Yao, Y. Liu, Fast evolution strategies, in: International Conference on Evolutionary Programming, Springer, 1997, pp. 149–161.
[26] H.-P.P. Schwefel, Evolution and Optimum Seeking: The Sixth Generation, John Wiley & Sons Inc., 1993.
[27] J. Cheng, Z. Pan, H. Liang, Z. Gao, J. Gao, Differential evolution algorithm with fitness and diversity ranking-based mutation operator, Swarm Evol. Comput. 61 (2021) 100816.
[28] L. Gui, X. Xia, F. Yu, H. Wu, R. Wu, B. Wei, Y. Zhang, X. Li, G. He, A multi-role based differential evolution, Swarm Evol. Comput. 50 (2019) 100508.
[29] S. Rahnamayan, H.R. Tizhoosh, M.M. Salama, Opposition-based differential evolution, IEEE Trans. Evol. Comput. 12 (1) (2008) 64–79.
[30] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.
[31] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2008) 398–417.
[32] N.Q. Huy, O.Y. Soon, L.M. Hiot, N. Krasnogor, Adaptive cellular memetic algorithms, Evol. Comput. 17 (2) (2009) 231–256.
[33] W. Jakob, A general cost-benefit-based adaptation framework for multimeme algorithms, Memet. Comput. 2 (3) (2010) 201–218.
[34] Y.S. Ong, A.J. Keane, Meta-lamarckian learning in memetic algorithms, IEEE Trans. Evol. Comput. 8 (2) (2004) 99–110.
[35] J. Yang, J. Yu, C. Huang, Adaptive multistrategy ensemble particle swarm optimization with signal-to-noise ratio distance metric, Inform. Sci. 612 (2022) 1066–1094.
[36] H. Liu, X.-W. Zhang, L.-P. Tu, A modified particle swarm optimization using adaptive strategy, Expert Syst. Appl. 152 (2020) 113353.
[37] S. Duman, H.T. Kahraman, B. Korkmaz, H. Bakir, U. Guvenc, C. Yilmaz, Improved phasor particle swarm optimization with fitness distance balance for optimal power flow problem of hybrid ac/dc power grids, in: The International Conference on Artificial Intelligence and Applied Mathematics in Engineering, Springer, 2021, pp. 307–336.
[38] T. Li, J. Shi, W. Deng, Z. Hu, Pyramid particle swarm optimization with novel strategies of competition and cooperation, Appl. Soft Comput. 121 (2022) 108731.
[39] Z. Meng, Y. Zhong, G. Mao, Y. Liang, PSO-sono: A novel PSO variant for single-objective numerical optimization, Inform. Sci. 586 (2022) 176–191.
[40] T.M. Shami, S. Mirjalili, Y. Al-Eryani, K. Daoudi, S. Izadi, L. Abualigah, Velocity pausing particle swarm optimization: A novel variant for global optimization, Neural Comput. Appl. 35 (12) (2023) 9193–9223.
[41] D. Zhu, J. Shen, Y. Zhang, W. Li, X. Zhu, C. Zhou, S. Cheng, Y. Yao, Multi-strategy particle swarm optimization with adaptive forgetting for base station layout, Swarm Evol. Comput. 91 (2024) 101737.
[42] P. Bujok, P. Kolenovsky, Eigen crossover in cooperative model of evolutionary algorithms applied to CEC 2022 single objective numerical optimisation, in: 2022 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2022, pp. 1–8.
[43] V. Stanovov, S. Akhmedova, E. Semenkin, Nl-shade-lbc algorithm with linear parameter adaptation bias change for CEC 2022 numerical optimization, in: 2022 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2022, pp. 01–08.
[44] R. Biedrzycki, J. Arabas, E. Warchulski, A version of nl-shade-rsp algorithm with midpoint for cec 2022 single objective bound constrained problems, in: 2022 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2022, pp. 1–8.
[45] P. Kolenovsky, P. Bujok, An adaptive variant of jso with multiple crossover strategies employing eigen transformation, in: 2022 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2022, pp. 1–8.
[46] B. Sun, Y. Sun, W. Li, Multiple topology shade with tolerance-based composite framework for CEC 2022 single objective bound constrained numerical optimization, in: 2022 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2022, pp. 1–8.