Neural Computing and Applications
https://doi.org/10.1007/s00521-021-06849-z

ORIGINAL ARTICLE

A hybrid cooperative differential evolution assisted by CMA-ES with local search mechanism

Fuqing Zhao[1] · Haizhu Bao[1] · Ling Wang[2] · Xuan He[3] · Jonrinaldi[4]

Received: 10 February 2021 / Accepted: 12 December 2021

Abstract
In this paper, a hybrid cooperative differential evolution with the perturbation of the Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) with the local search of Limited-Memory Broyden–Fletcher–Goldfarb–Shanno (LBFGS) mechanism, named jSO_CMA-ES_LBFGS, is proposed to solve the complex continuous problems. In the proposed algorithm, jSO, as a variant of Differential Evolution (DE), is used as a global search operator to explore the entire solution space. When the population falls into stagnation, a relatively reliable initial solution for the local search operator is generated by the CMA-ES, which is activated to perturb the optimal candidates in the solution space. The LBFGS utilized as the local search strategy is embedded in CMA-ES to obtain the potential local optimal solutions. A cooperative co-evolutionary dynamic system is formed by jSO and CMA-ES with a local search operator. The proposed jSO_CMA-ES_LBFGS is tested on the CEC2017 benchmark test suite and compared with eleven state-of-the-art algorithms. Further, two practical engineering problems are investigated utilizing the proposed method. The experimental results reveal the effectiveness and efficiency of the jSO_CMA-ES_LBFGS.

Keywords Differential Evolution (DE) · Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) · Limited-Memory Broyden–Fletcher–Goldfarb–Shanno (LBFGS) · Cooperative coevolution · Numerical optimization

✉ Fuqing Zhao
zhaofq@lut.edu.cn

Haizhu Bao
zhangjl_lut@lut.edu.cn

Ling Wang
wangling@tsinghua.edu.cn

Xuan He
hexuan@shu.edu.cn

Jonrinaldi
jonrinaldi@eng.unand.ac.id

[1] School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China

[2] Department of Automation, Tsinghua University, Beijing 10084, China

[3] School of Mechatronic Engineering and Automation, Shanghai University, Shanghai 200444, China

[4] Department of Industrial Engineering, Universitas Andalas, Padang 25163, Indonesia

1 Introduction

As an important branch of artificial intelligence, evolutionary computation, which has the characteristics of self-adaptation, self-organization, and self-learning, is a kind of model and algorithm for simulating the behavior of "the survival of the fittest" in natural biological and ecological systems, and the algorithm is used to solving various complicated problems [1, 2]. The continuous optimization problem from the real world is an important application field of evolutionary computation [3, 4]. It has been an important issue and has been widely studied in recent years. Classical evolutionary algorithms mainly include genetic algorithm (GA) [5], differential evolution [6], artificial bee colony (ABC) [7], particle swarm optimization (PSO) [8], whale optimization algorithm (WOA) [9], and the other typical hybrid evolutionary computation algorithms.

The DE is proposed to find the global optimum solution for complex optimization problems [10]. The basic idea of DE stems from Darwin's theory of biological evolution.

Owing to the efficiency and effectiveness of solving optimization problems, DE is applied to solve numerous real-world problems from diverse domains of engineering optimization, e.g., electrical and power systems [11], manufacturing science, and operation research [12, 13]. It is difficult to accurately classify the methods of DE since certain variants of DE are combinations of multiple mechanisms. The modifications of DE are roughly divided into three categories, and the details are shown as follows.

(i) DE with certain strategies.

A variant of DE with multi-population and mutation strategies (EDEV) is proposed by Wu et al. [14], which utilized the combination of different evolution strategies and different subpopulations to produce the next generation solution in different evolution stages. The surrogate is embedded as a strategy in DE to predict the optimal solution and guide the search direction of differential evolution [15, 16]. The experimental results verify that the surrogate helps speed up the convergence of DE. In [17], the stochastic mutation strategy and information sharing mechanism are introduced to enhance the search capability of DE. In addition, incorporating a local search heuristic is often useful in enhancing the search performance of DE [18].

(ii) DE with Adaptation of the Parameters.

The performance of DE depends on its control parameters and search operators. In DE, the values of two parameters are automatically configured and only one mutation strategy is used throughout the evolution process [19]. The evolution process of the algorithm is recognized as a closed-loop control system. In [20], two sinusoidal waves are used to adapt to the scaling factor. A performance adaptation scheme based on earlier success is used to choose one of the sinusoidal waves. Moreover, the covariance matrix learning strategy is used for the crossover operator to solve problems with a high correlation between the variables. In [21], a parameter with the adaptive learning mechanism (PALM) is introduced to tackle the ill interaction of control parameters in certain DE variants, and a timestamp mechanism is employed to update inferior solutions in the external archive.

(iii) Hybridization of DE with other algorithms.

In the domain of optimization, hybridization refers to the combination of different algorithms to achieve the effect of "chemical reaction" according to the optimization characteristics of the algorithm, rather than the simple flattening of the different operating mechanisms of the algorithms. In [22], the combination of ABC and DE, which is named HABCDE, is proposed to develop a more effective algorithm than ABC and DE in terms of convergence and exploration–exploitation. Moreover, in [23], DE is used as a local search process in the moth search algorithm to enhance local searchability. The experimental performance analysis confirms that the hybrid mechanism is effective for solving practical optimization problems. In [24], a hybrid system of DE and bat evolution algorithm (BA), in which DE and BA are in a competitive relationship, is introduced to balance the exploration and exploitation capabilities of BA.

According to the literature, since Zhang and Sanderson [25] proposed a variant of DE by implementing the mutation strategy "$\mathrm{DE/current-to-pbest}$" with the external archive $\boldsymbol{A}$ to improve optimization performance (JADE), there has been a variety of researches focusing on adaptation of the control parameters of DE. The feedback from the process of evolutionary search is employed to dynamically tune the value of control parameters. By adopting certain self-learning mechanism, the control parameters of DE are automatically adjusted to appropriate values. An improved JADE, which is named as SHADE [22], is proposed to improve the robustness of JADE.

A hybridization framework of LSHADE-SPACMA based on the LSHADE and CMA-ES is introduced to improve the optimization performance of L-SHADE algorithm [26]. DE as a local search operator is introduced into the Grey Wolf Optimizer (GWO) to improve the exploitation of the GWO (COGWO2D) [27]. The experimental results proved that COGWO2D is effective to find the optimal solutions of the global optimization problem. Adaptive neighborhood operators are borrowed from DE to promote exploration and exploitation of Migrating Birds Optimization (MBO) [28].

In summary, DE has been extensively studied on the operating mechanism of the algorithm regarding whether it is a single DE algorithm or a hybrid DE algorithm in recent years. However, DE tends to trap into the local optima and lose the diversity of the population in the evolution process. First, the size of the external archive is fixed in the classic evolution strategy of DE. As the evolution of the population progresses, the solutions in the external archive are constantly replaced by updated candidates with optimal fitness. The population falls into local optimal solutions, the external archive stores various suboptimal solutions rather than worse solutions. In other words, the perturbation strength is insufficient currently. Second, according to the evolutionary state of the population, the strength of the disturbance is automatically adjusted rather than a predetermined single value. Third, to enhance the local search ability of the hybrid algorithm for DE, the population of DE will be degraded by injecting results of local optimization.

Inspired by the cooperative co-evolutionary framework, a hybrid cooperative jSO [19] (a optimization algorithm for solving single-objective real parameter problems) with the perturbation of CMA-ES with a local search of LBFGS, named jSO_CMA-ES_LBFGS, is proposed to solve the single objective numerical optimization problems. In the jSO_CMA-ES_LBFGS, the jSO provides a reliable initial solution for CMA-ES with LBFGS, and CMA-ES with LBFGS provides a potentially local optimal solution for jSO, which makes jSO have a wide range of learning capabilities. Therefore, a cooperative co-evolution between jSO and CMA-ES with LBFGS is achieved. The contributions of this paper are generalized as follows.

(1) In this paper, a hybrid cooperative differential evolution algorithm is designed to solve the single-objective real-parameter numerical optimization problem, which effectively combines evolutionary algorithm (jSO), an evolutionary strategy (CMA-ES), and mathematical method (LBFGS). A de-randomized and anti-rotation CMA-ES [29] is used as a perturbation operator to provide a relatively reliable initial solution for the local search operator (LBFGS).
(2) The Markov model of jSO_CMA-ES_LBFGS is presented to analyze the convergence performance of the jSO_CMA-ES_LBFGS mathematically. The evolutionary process of the jSO_CMA-ES_LBFGS is mapped to the state transition process in the Markov model, and then the convergence performance of the jSO_CMA-ES_LBFGS is proved.
(3) The proposed jSO_CMA-ES_LBFGS is tested on CEC 2017 benchmarks and compared with the seven variants of DE and other evolutionary algorithms. The experimental results imply that the jSO_CMA-ES_LBFGS obtain more efficient and accurate solutions than that of the other algorithms. Furthermore, a desirable combination of parameters is also analyzed by executing the Taguchi method [30].

The remainder of the paper is organized as follows. The framework of the original DE and CMA-ES is summarized in Sect. 2. The proposed methodology is described and analyzed in Sect. 3. The experimental research and statistical analysis are provided in Sect. 4. In Sect. 5, two practical engineering problems including continuous and discrete decision space are investigated using the proposed jSO_CMA-ES_LBFGS. Finally, the conclusions and future work are presented in Sect. 6.

2 Basic algorithm and operators

In this section, jSO and CMA-ES are briefly introduced. The notation is described as follows.

2.1 jSO algorithm

The basic operations of the DE include mutation, crossover, and selection. When the population of DE is initialized, three basic operations are repeated until a certain termination criterion (e.g., exhaustion of maximum functional evaluations) is satisfied. In the DE, the population is a set of real value vectors $x_i = (x_1, \ldots, x_D),\ \ i = 1, \ldots, NP$, where D is the dimension of the objective function, and NP is the size of the population. The population is initialized randomly.

The mutation operator:

$$v_{i,g} = x_{i,g} + F_w\left(x_{pBest,g} - x_{i,g}\right) + F\left(x_{r1,g} - x_{r2,g}\right) \tag{1}$$

where F is scale factor and F_w is calculated as follows:

$$F_w = \begin{cases} 0.7 * F, nfes < 0.2Max\_FEs, \\ 0.8 * F, nfes < 0.4Max\_FEs, \\ 1.2 * F, otherwise \end{cases} \tag{2}$$

The indexes r_1, r_2 are chosen from $[1 : N + NP]$ randomly, where N is the size of the external archive $\boldsymbol{A}$ and $r_1 \neq r_2 \neq i$. $x_{pBest,g}$ are the dominant individuals in the population. The parameter $F \epsilon [0 : 1]$ is the scaling factor of the i-th individual, which controls the size of the difference term. If the donor vector goes beyond the feasible region of the problem, the following operation is performed:

$$v_{i,j,g} = \begin{cases} \left(x_j^{min} + x_{i,j,g}\right)/2 & if \quad v_{i,j,g} < x_j^{min} \\ \left(x_j^{max} + x_{i,j,g}\right)/2 & if \quad v_{i,j,g} < x_j^{max} \end{cases} \tag{3}$$

The crossover operator:

$$\mu_{i,j,g} = \begin{cases} v_{i,j,g} & if\ r\ and\ [0,1) > CR\, or\, j = j_{rand}, \\ x_{i,j,g} & otherwise. \end{cases} \tag{4}$$

The selection operator:

$$x_{i,g} = \begin{cases} u_{i,g} & if\, f\left(u_{i,G}\right) \le f\left(x_{i,G}\right) \\ x_{i,g} & otherwise \end{cases} \tag{5}$$

The jSO algorithm is one of the most advanced variants of the DE family [19, 31]. The jSO, with the new weighted version of the mutation strategy, inherits the evolutionary strategy and parameter control mechanism of the L-SHADE. The parameter control mechanism of jSO is an adaptive closed-loop control system. The values of the control parameters of jSO are different at different stages of evolution, which embodies a search process from global to local.

2.2 CMA-ES

A de-randomized evolutionary strategy with the covariance matrix adaptation (CMA-ES) is used as a perturbation operator to avoid using a pre-determined single value for the perturbation strength [29]. A standard CMA-ES consists of the three control parameters, including mean vector m, covariance matrix C, and step-size σ. m is the favorite solution. σ is the control step length. C is the shape of the distribution ellipsoid. The adaptive update process for these three parameters is as follows.

NP	The population size of the jSO
D	Dimension size
$f(X)$	The object function
g	Generation counter
$x_{pBest,g}$	$x_{pBest,g}$ is randomly chosen as one of the top 100 $p\%$ individuals in the current population with $P \in [0.25, 0.125]$
$\boldsymbol{A}$	External archive
$X_{best}^{current}$	The best solution of current population
num_g	The number of times the g-th generation population has stagnated
σ	The so-called step-size $\sigma \in R_+$ controls the step length
m	The mean vector $m \in R^n$ represents the favorite solution
C	The covariance matrix $C \in R^{n\times n}$ determines the shape of the distribution ellipsoid
N_i	The multi-variate normal distribution
p_c	The evolution path of C
p_σ	The evolution path of σ
λ	The population size of CMA-ES, default value:$\lambda = 4 + round(3.0 * \log(D))$
X_{best}	The best solution obtained by the proposed algorithm
H_k	Hessian matrix of the objective function in the k-th iteration of LBFGS
∇f_k	The first derivative of the objective function in the k-th iteration of LBFGS
EFs	The number of evaluations

The search solutions are sampled by a normally distributed.

$$x_i = m + \sigma y_i, y_i \sim N_i(0, C) \tag{6}$$

The mean is updated.

$$m \leftarrow \sum_{i=1}^{\mu} w_i x_{i:\lambda} = m + \sigma y_w \tag{7}$$

where $y_w = \sum_{i=1}^{\mu} w_i y_{i:\lambda}$. The evolutionary path of C is constructed and accumulated.

$$p_c \leftarrow (1 - c_c)p_c + \sqrt{1 - (1 - c_c)^2}\sqrt{\mu_w} y_w \tag{8}$$

The evolutionary path of σ is constructed and accumulated.

$$p_\sigma \leftarrow (1 - c_\sigma)p_\sigma + \sqrt{1 - (1 - c_\sigma)^2}\sqrt{\mu_w} C^{-\frac{1}{2}} y_w. \tag{9}$$

The covariance matrix C is updated.

$$C \leftarrow (1 - c_1 - c_\mu)C + c_1 p_c p_c^T + c_\mu \sum_{i=1}^{\mu} w_i y_{i:\lambda} y_{i:\lambda}^T. \tag{10}$$

The step length σ is updated.

$$\sigma \leftarrow \sigma \times exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{||p_\sigma||}{E||N(0,I)||} - 1\right)\right) \tag{11}$$

In summary, the main characteristics of CMA-ES are as follows: (i) Multivariate normal distribution is adopted to generate search solutions and conforms to the maximum entropy principle, in which the least possible assumptions on target function are used in the distribution shape. (ii) A rank-based selection strategy implies rotational invariance. (iii) The convergence rate is improved by a step-size control mechanism. (iv) Covariance matrix adaptation (CMA) increases the likelihood of previously successful steps, which reduces any convex-quadratic function to the sphere model without using derivatives. Please refer to the mentioned literature which includes a further description of the CMA-ES.

3 The description of the proposed algorithm

The three problems mentioned earlier in Sect. 1 about DE are critical reasons that restrict the improvement of the algorithm. Firstly, for the problem of disturbance intensity mentioned in the first problem and the second problem, CMA-ES is utilized as a disturbance operator, so that the parameters in the evolution process of the algorithm change adaptively and dynamically. Secondly, the designed stagnation detection operator is used to detect the state of the population all the time, and different operations are activated according to different states, to ensure the self-cooperation between algorithm operations as far as possible. For the third problem, a mathematical method (LBFGS) is employed to fuse in the local search stage of the evolutionary algorithm to conduct in-depth exploitation. In Sects. 3.1 and 3.2, the importance of the LBFGS in the jSO_CMA-ES_LBFGS is proved by detailed experiments. In addition, the effect of ensemble operators has been proved by experiments in Sect. 4.4.

As shown in Fig. 1, firstly, the jSO searches the entire search space as a global search operator. In the process, the stagnation detection operator is performed to detect the

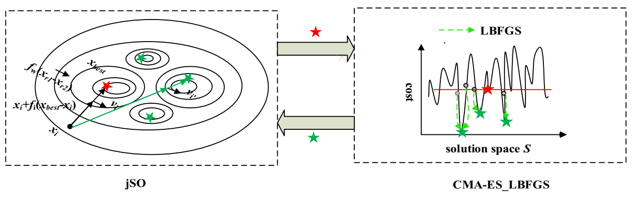

Fig. 1 The operating mechanism of jSO_CMA-ES_LBFGS

state of the population. The perturbation operator is activated when the population falls into a locally optimal solution (red pentagram). Second, the local optimal solution generated by jSO is considered as the mean vector parameter of the perturbation operator (CMA-ES). New λ search points, which distributes around the local optimal solution (red line), are generated by a normally distributed (gray spot). Furthermore, a local search (LBFGS) is performed on the λ search points to obtain the potential local optimal solutions (green pentagram). Finally, these local optimal solutions guide the population of jSO out of the current attraction basin to explore another area containing a potential local optimal solution.

The jSO provides the CMA-ES_LBFGS with a reliable initial solution, while CMA-ES_LBFGS returns multiple local optimal solutions to jSO. Then, jSO learns from these multiple local optimal solutions, thus jumping out of the current attraction basin. Therefore, the cooperative evolution between jSO and CMA-ES_LBFGS is achieved.

Algorithm 1. Pseudo-code of the jSO_CMA-ES_LBFGS.

```
Input: $f(X)$(the objective function), $D$(dimension size)
Output: $X_{best}$ (the best solution)
1: Initialize parameters
2: While stop criterion is not satisfied do
3:  Exploration phase using jSO with modified mutation strategy by utilizing Algorithm 3.
4:  Stagnation detection operator is executed.
5:  If the current population is stagnant
6:  $\mathrm{CMA-ES}_{\mathrm{LBFGS}_{pop}} \leftarrow \mathrm{CMA-ES\ with\ LBFGS}(X_{best}^{current}, \sigma)$
    //update the $\mathrm{CMA-ES}_{\mathrm{LBFGS}_{pop}}$ by Algorithm 2.
7:  End If
8: End While
```

The pseudocode of jSO_CMA-ES_LBFGS is presented in Algorithms 1–3. As the LBFGS performs a local search on a copy of the population of CMA-ES, LBFGS does not affect the adaptability and convergence of CMA-ES. At the same time, the local optimization results are not directly injected into the jSO. The adaptability and convergence of jSO are not affected by CMA-ES and LBFGS. The size of the perturbation is provided by the CMA-ES, which explores the neighborhood of the current optimal solution for the jSO.

Algorithm 2. Pseudo-code of the CMA-ES with LBFGS.

```
Input: $m \in \boldsymbol{R}^n, \sigma \in \boldsymbol{R}_+$
Output: $\mathrm{CMA-ES}_{\mathrm{LBFGS}_{\mathrm{pop}}}$
1: While stop criterion is not satisfied do
2:  For $i = 1{:}\lambda$ do
3:    $x_i = m + \sigma y_i, y_i \sim N_i(0, \boldsymbol{C})$
4:    $x_i' \leftarrow LBFGS(x_i)$
5:    $EFs = EFs + 4$
6:    $\mathrm{CMA-ES\_LBFGS\_}pop(i) \leftarrow x_i'$
7:  End For
8:  The $m$, $C$, and $\sigma$ are updated by using Eq. (7)- Eq. (11).
9: End While
```

Algorithm 3. Pseudo-code of the jSO.

```
1: Initialize parameters ($M_F = 0.5$, $M_{CR} = 0.8$, $NP$)
2: Randomly initialize population ($P_g = (x_{i,g}, \dots, x_{NP,g})$)
3: Set index counter $k = 1$, set the archive $A \leftarrow \emptyset$.
4: While stop criterion is not satisfied do
5:  $S_{CR} \leftarrow \emptyset$, $S_F \leftarrow \emptyset$
6:  For $i = 1{:}NP$ do
7:    Randomly select from $[1, H]$ as $r_i$.
8:    If $r_i = H$ then
9:      $M_{F,r_i} = 0.9, M_{CR,r_i} = 0.9$
10:   End If
11:   If $M_{CR,r_i} < 0$ then $CR_{i,g} = 0$
12:   Else $CR_{i,g} \leftarrow N_i(M_{CR,r_i}, 0.1)$
13:   End If
14:   If $g < 0.25G_{max}$ then
        $CR_{i,g} = \max\{CR_{i,g}, 0.7\}$
15:   Else If $g < 0.5G_{max}$ then
        $CR_{i,g} \leftarrow \max\{CR_{i,g}, 0.6\}$
16:   End If
17:   $F_{i,g} \leftarrow C_i(M_{F,r_i}, 0.1)$
18:   If $g < 0.6G_{max}$ and $F_{i,g} < 0.7$ then
19:   $F_{i,g} = 0.7$
20:   End If
21:   Set $u_{i,g}$ by using Eq. (1)
22:  End For
23: For $i = 1{:}NP$ do
24:  If $f(u_{i,g}) \le f(x_{i,g})$ then
       $x_{i,g+1} = u_{i,g}, A \leftarrow x_{i,g}, S_{CR} \leftarrow CR_{i,g},\ S_F \leftarrow F_{i,g}$
25:  Else $x_{i,g+1} = x_{i,g}$
26:  End If
27:  $EFs = EFs + 2$
28:  Update $M_F$ and $M_{CR}$
29: End For
30: $g = g + 1$
31: End While
```

3.1 Stagnation detection operator

To measure the diversity of the population of jSO_CMA-ES_LBFGS, the diversity metric DP is defined as follows:

$$DP_g = \frac{1}{NP} \cdot \left(\sum_{i=1}^{NP} \left| x_{i,g} - \frac{1}{NP} \cdot \sum_{j=1}^{NP} x_{j,g} \right| \right), \tag{12}$$

$$num_{g+1} = \begin{cases} num_g + 1 & if \left| DP_{g+1} - DP_g \right| \sigma \cdot D \\ 0 & \text{Otherwise} \end{cases} \tag{13}$$

where DP_g is a metric that is used to measure the diversity of the g-generation population. D is the dimension size. The σ is the step size parameter in CMA-ES. The num_G is an indicator to detect whether the g-generation population is in a state of stagnation. $num_1 = 0$, if $num_g > T$, the population of jSO_CMA-ES_LBFGS is considered to be in a stagnant state and the perturbation operator (CMA-ES) is activated. Here, $T = 10$, the stagnation detection operator is executed to detect the state of the population of jSO. As shown in Figs. 2 and 3. The dynamics of jSO_CMA-ES_LBFGS in terms of population state (stagnation, explorative, and exploitative) are studied for f_4 and f_{11} functions to show the role of the stagnation detection operator.

3.2 Local search

The local search methods for solving unconstrained numerical optimization problems are mainly divided into two categories: iterative gradient decrement algorithm and random local search algorithm. Random local search algorithms mainly include Multi-Trajectory Local Search algorithm (Mtsls1), Solis and Wets' Algorithm (SW), etc.

The SW is a local search algorithm proposed by Solis and Wets [32]. The basic principle is to give an initial solution and randomly generate a possible solution in a normal distribution. In the search process, according to the increase or decrease in the value of the objective function, the direction and step size of the search are adjusted to find the optimal solution. A candidate solution requires 3 times functional evaluations in the SW algorithm.

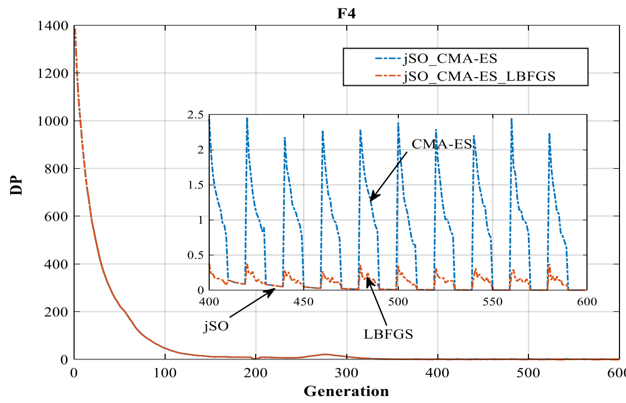

Fig. 2 The diversity of the population of jSO_CMA-ES_LBFGS (f_4, $D = 30$)

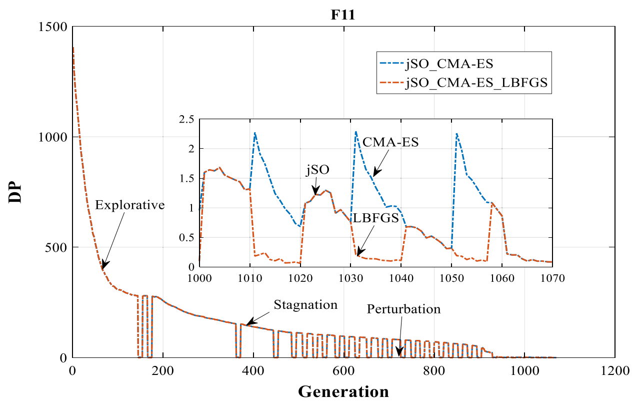

Fig. 3 The diversity of the population of jSO_CMA-ES_LBFGS (f_{11}, $D = 30$)

The Mtsls1 is a well-known local search algorithm [33]. As shown in Fig. 4, the Mtsls1 searches the solution space by dimension. A solution starts from the first dimension and ends with the last dimension. Furthermore, it is updated one by one.

Move step size s along one dimension, if the fitness value is reduced, then obtained solution is kept for the next dimension (from solution a to solution b). If the fitness value increases, the Mtsls1 returns to the starting solution and moves by $0.5 \times s$ along the reverse direction (from solution b to solution c). If the fitness of solution c is smaller than that of a, solution c is kept as the starting solution in the next dimension. Therefore, the update process of the Mtsls1 is greedy.

The LBFGS belongs to an iterative gradient decrement algorithm. LBFGS is known for the fast convergence for solving the convex optimization problem and achieves higher accuracy with less number of function evaluations [34]. The core idea behinds the Broyden–Fletcher–

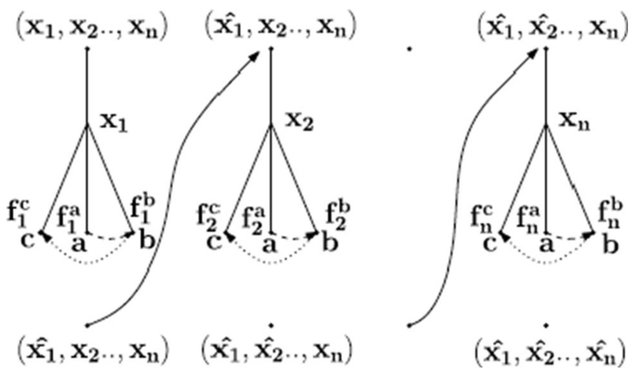
$(x_1, x_2.., x_n)$
$(\hat{x}_1, x_2.., x_n)$
$(\hat{x}_1, \hat{x}_2.., x_n)$
x_1
x_2
x_n
f_1^c f_1^a f_1^b
f_2^c f_2^a f_2^b
f_n^c f_n^a f_n^b
c a b
$(\hat{x}_1, x_2.., x_n)$
$(\hat{x}_1, \hat{x}_2.., x_n)$
$(\hat{x}_1, \hat{x}_2.., \hat{x}_n)$

Fig. 4 Mtsls1 search along one dimension from the first dimension to the last dimension (Adapted from [31])

Goldfarb–Shanno (BFGS), without using second-order partial derivatives of the function, is to construct a positive definite symmetric matrix that approximates the Hessian matrix. The objective function is optimized by BFGS under the condition of "quasi-Newton".

Each step of the BFGS method has the form,

$$d_k = -H_k \nabla f_k \tag{14}$$

$$x_{k+1} = x_k + \alpha_k d_k \tag{15}$$

where α_k is the step length and α_k satisfies the Wolfe conditions:

$$f(x_k + \alpha_k d_k) \le f(x_k) + \beta' \alpha_k (\nabla f_k)^T d_k, \tag{16}$$

$$(\nabla f_{k+1})^T d_k \ge \beta (\nabla f_k)^T d_k, \tag{17}$$

$$0 < \beta' < \frac{1}{2}, \beta' < \beta < 1. \tag{18}$$

Moreover, if $\alpha_k = 1$ satisfies Eq. (16), Eq. (17), and Eq. (18), $\alpha_k = 1$. H_k is updated at every iteration utilizing the formula,

$$H_{k+1} = V_k^T H_k V_k + \rho_k s_k s_k^T \tag{19}$$

where

$$\rho_k = \frac{1}{y_k^T s_k}, \tag{20}$$

$$V_k = I - \rho_k y_k s_k^T, \tag{21}$$

$$s_k = x_{k+1} - x_k, \tag{22}$$

$$y_k = \nabla f_{k+1} - \nabla f_k. \tag{23}$$

The information from the recent iterations is utilized to construct the approximate Hessian in the LBFGS [35–37]. The termination condition of LBFGS is satisfied when the candidates move one step in the search space. According to the reference manual of ALGLIB and the user guidance, LBFGS uses a 4-point central formula for differentiation. Therefore, one gradient calculation requires $4 \times D$ times function evaluations, where D is dimension size. Because the gradient is the fastest direction in which the function value decreases, LBFGS is adopted as a local search operator to achieve the fast convergence speed of the proposed algorithm.

3.3 Principle and convergence analysis of jSO_CMA-ES_LBFGS

In this paper, the results of local optimization are not directly injected into jSO. Therefore, the adaptability and convergence of jSO are not affected by CMA-ES and LBFGS. The convergence of jSO proved to be equivalent to the convergence of jSO_CMA-ES_LBFGS.

As a population-based stochastic optimization algorithm, the evolutionary process of jSO is considered to be a stochastic process. Let $S = R^D$ be the individual solution space, S^{NP} denote the population space, and $f : S \to R^+$ be the fitness function, where D is the dimension of the problem and NP is the population size. The basic operators of DE are described as follows.

The stochastic mapping of the mutation operator:

$$T_m : S^{NP} \to S \tag{24}$$

The probability distribution of the mutation operator:

$$\begin{aligned} &P\{T_m(X) = v_i\} = \sum P\left(T_m^1(x)\{x_{r1}, x_{r2}, x_{i,G}, x_{pBest,g}\}\right) \cdot \\ &P\left(T_m^2(x_{r1}, x_{r2}, x_{i,G}, x_{pBest,g}) = v_i\right) \\ &= \sum P\left(T_m^1(x) = \{x_{r1}, x_{r2}, x_{i,G}, x_{pBest,g}\}\right), \\ &\{x_{r1}, x_{r2}, x_{i,G}, x_{pBest,g}\} \in S^4. \end{aligned} \tag{25}$$

The stochastic mapping of the crossover operator:

$$T_c : S^2 \to S \tag{26}$$

The probability distribution of the crossover operator:

$$\mathrm{P}\{T_c(x_i, v_i) = u_i\} = \begin{cases} 0, & u_i = x_i \\ m \cdot C_D^k CR^k (1 - CR)^{D-k}, & else \\ CR^N + \dfrac{1}{NP}, & u_i = v_i \end{cases} \tag{27}$$

The stochastic mapping of the selection operator:

$$T_s : S^2 \to S. \tag{28}$$

The probability distribution of the selection operator:

$$P\{T_s(x_i, u_i) = u_i\} = \begin{cases} 1, & f(u_i) \le f(x_i) \\ 0, & else \end{cases} \tag{29}$$

From the above analysis, jSO algorithm is written as

$$X(n+1) = \{x_i(n+1) = T_s^\circ T_c^\circ T_m(X(n)), i = 1, \cdots NP\} \tag{30}$$

Definition 1 (*Convergence in Probability*) [38] Let $\{X(n), n = 0, 1, 2 \ldots\}$ be a population sequence generated by a population-based stochastic algorithm, the stochastic sequence $\{X(n)\}$ weakly converges in probability to the global optimum. If and only if:

$$\lim_{n \to \infty} P\{X(n) \cap B^* \neq \emptyset\} = 1 \tag{31}$$

where B^* is a set of global optimum of an optimization problem.

Lemma 1 *In the jSO algorithm, the evolutional direction of the population is monotonically non-increasing, i.e.,* $F(X(n+1)) \le F(X(n))$.

Lemma 2 *The population sequences of DE algorithm $X(n), n \in N^{+}$ form a Markov chain process.*

Proof The population sequences of jSO are.

$$X(n+1) = T(X(n)) = T_s^{\circ} T_c^{\circ} T_m(X(n)), \tag{32}$$

where the T_s, T_c, and T_m are irrelevant to the iteration n and $X(n+1)$ only depends on $X(n)$. The transition probability of states is calculated by

$$\begin{aligned} &P\{T(X(n))_i = x_i(n+1)\} \\ &= \sum\sum\sum P\{T_m^1(X(n)) = \{x_{r1}, x_{r2}, x_{i,G}, x_{pBest,g}\}\} \\ &\quad \cdot P\{T_c(x_i(n), v_i) = u_i\} \\ &\quad \cdot P\{T_s(x_i(n), u_i) = x_i(n+1)\}, \\ &\quad u_i \in S, v_i \in S, \{x_{r1}, x_{r2}, x_{i,G}, x_{pBest,g}\} \in S^4. \end{aligned} \tag{33}$$

It is obvious that $\forall X(n) \in S^{NP}, \exists\{x_{r1}, x_{r2}, x_{i,G}, x_{pBest,g}\} \in S^{NP}$ and $v_i, u_i \in S^{NP}$, subject to

$$P\{T_m^1(X(n)) = \{x_{r1}, x_{r2}, x_{i,G}, x_{pBest,g}\}\} > 0 \tag{34}$$

$$P\{T_m^2(x_{r1}, x_{r2}, x_{i,G}, x_{pBest,g}) = v_i\} > 0, \tag{35}$$

$$P\{T_c(x_i(n), v_i) = u_i\} > 0, \tag{36}$$

$$P\{T_s(x_i(n), u_i) = x_i(n+1)\} > 0 \tag{37}$$

So,

$$P\{T(X(n))_i = x_i(n+1)\} > 0, \tag{38}$$

not depending on n. Thus, the transition probability of states is given as

$$P\{T(X(n)) = X(n+1)\} = \prod_{i=1}^{NP} P\{T(X(n))_i = x_i(n+1)\} > 0. \tag{39}$$

This transition probability is also independent of n. Therefore, the population sequence of DE is a homogeneous irreducible aperiodic Markov chain. Markov population sequence of jSO is represented by

$$P\{X, Y\} = P\{X(n+1) = Y | X(n) = X\} \tag{40}$$

□

Theorem 1 *Suppose that $\{x(t), t = 0, 1, 2, \ldots\}$ is the population sequence generated by DE, then, $\{x(t), t = 0, 1, 2, \ldots\}$ converges to the subset $B_0^* = \{Y = (y_1, \cdots y_{NP}); y_i \in B^*\}$ of global optimum B^* with probability one, i.e.*

$$\lim_{n\to\infty} P\{X(n) \in B_0^* | X(0) = X_0\} = 1 \tag{41}$$

Proof Suppose x^* is the unique optimum satisfaction solution. The following properties are derived according to Eq. (25) and Eq. (27).

(1) if X, Y $\in B_0^*$, then $P\{X, Y\} > 0$, $P\{Y, X\} > 0$, in which these two states are interconnected; i.e.,$X \leftrightarrow Y$.
(2) if X $\in B_0^*$ and Y $\notin B_0^*$ then $P\{X, Y\} = 0$, i.e.,$X \not\leftrightarrow Y$.

Hence B_0^* is a positive recurrent, irreducible, aperiodic, and closed set. According to the properties of the aperiodic, homogeneous Markov chain [39], the sequence $\{x(t), t = 1, 2, \ldots,\}$ exists a limiting distribution $\pi(Y)$,

$$\lim_{n\to\infty} P\{X(n) = Y | X(0) = X_0\} = \begin{cases} \pi(Y), & Y \in B_0^* \\ 0, & otherwise. \end{cases} \tag{42}$$

Then,

$$\lim_{t\to\infty} P\{X(n) = B_0^*\} = 1 \tag{43}$$

So,

$$\lim_{n\to\infty} P\{X(n) \cap B^* \neq \emptyset\} = 1 \tag{44}$$

According to Definition 1, we can obtain that jSO converges to the global optimum in probability.

□

4 Experiments and comparisons

The performance of the proposed algorithm and the comparison algorithm is evaluated in this section. Firstly, a brief introduction of the CEC2017 benchmark function is shown as follows.

The CEC2017 benchmark test suite contains 30 test functions, $f_1 - f_2$ are unimodal functions, $f_4 - f_{10}$ are simple multimodal functions, $f_{11} - f_{20}$ are hybrid functions, and $f_{21} - f_{30}$ are composition functions. These functions are classic, representative single-objective continuous optimization functions, which are all parameter optimization functions. They are published by an expert group of the special theme of the annual CEC conference to test the current optimization methods (especially the Meta-heuristic optimization method). The summary of the CEC2017 test function is shown in Table 1. The last column in the table represents the optimal value of each function. f_2 has been excluded because it shows unstable behavior especially for higher dimensions. There are 20 basic functions [40] defined to construct the CEC2017 test function. The unimodal functions and simple multimodal functions are obtained by adding rotation and translation operations to

Table 1 Summary of the CEC2017 test function

Categorization	No	Functions	$F_i^* = F(x^*)$
Unimodal functions	f_1	Shifted and Rotated Bent Cigar Function	100
	f_2	Shifted and Rotated Sum of Different Power Function	200
	f_3	Shifted and Rotated Zakharov Function	300
Simple multimodal functions	f_4	Shifted and Rotated Rosenbrock's Function	400
	f_5	Shifted and Rotated Rastrigin's Function	500
	f_6	Shifted and Rotated Expanded Scaffer's F6 Function	600
	f_7	Shifted and Rotated Lunacek Bi_Rastrigin Function	700
	f_8	Shifted and Rotated Non-Continuous Rastrigin's Function	800
	f_9	Shifted and Rotated Levy Function	900
	f_{10}	Shifted and Rotated Schwefel's Function	1000
Hybrid functions	f_{11}	Hybrid Function 1 ($NF = 3$)	1100
	f_{12}	Hybrid Function 2 ($NF = 3$)	1200
	f_{13}	Hybrid Function 3 ($NF = 3$)	1300
	f_{14}	Hybrid Function 4 ($NF = 4$)	1400
	f_{15}	Hybrid Function 5 ($NF = 4$)	1500
	f_{16}	Hybrid Function 6 ($NF = 4$)	1600
	f_{17}	Hybrid Function 6 ($NF = 5$)	1700
	f_{18}	Hybrid Function 6 ($NF = 5$)	1800
	f_{19}	Hybrid Function 6 ($NF = 5$)	1900
	f_{20}	Hybrid Function 6 ($NF = 3$)	2000
Composition functions	f_{21}	Composition Function 1 ($NF = 3$)	2100
	f_{22}	Composition Function 2 ($NF = 3$)	2200
	f_{23}	Composition Function 3 ($NF = 4$)	2300
	f_{24}	Composition Function 4 ($NF = 4$)	2400
	f_{25}	Composition Function 5 ($NF = 5$)	2500
	f_{26}	Composition Function 6 ($NF = 5$)	2600
	f_{27}	Composition Function 7 ($NF = 6$)	2700
	f_{28}	Composition Function 8 ($NF = 6$)	2800
	f_{29}	Composition Function 9 ($NF = 3$)	2900
	f_{30}	Composition Function 10 ($NF = 3$)	3000

the basic functions. The hybrid functions and composition functions are obtained by embedding different basic functions into different subcomponents of variables, where *NF* is the number of basic functions.

In Sect. 4.1, the time complexity of the jSO_CMA-ES_LBFGS is analyzed. In Sect. 4.2, the experimental condition and parameters setting are described when the simulation is run. The role of the local search algorithm is tested in Sect. 4.3. The integration effect of different operators is provided in Sect. 4.4. The comparison of jSO_CMA-ES_LBFGS with certain representative algorithms is described in Sect. 4.5. Moreover, owing to limited space, interested readers obtain the experimental results of all algorithms from the online supplemental material.

4.1 jSO_CMA-ES_LBFGS time complexity

This subsection deals with the time complexity of the jSO_CMA-ES_LBFGS. The running time obtained by evaluating the benchmark function f_{18} is compared with the running time of the test program.

The computing time of the test program is denoted as T0. Let us assume that variable T1 presents the time required for evaluating the benchmark function f_{18} and variable T2 the time of jSO_CMA-ES_LBFGS execution for function f_{18} within $200,000$ evaluations for each dimension. Variable $\widehat{\mathrm{T2}}$ is an average of T2 values obtained in five independent runs. Table 2 shows the algorithm complexities relationship with dimension. The computational complexity of the algorithm jSO_CMA-ES_LBFGS is reflected by the measured and calculated

Table 2 The computational complexity of the algorithm jSO_CMA-ES_LBFGS (all times are in seconds)

D	$T0$	$T1$	$\widehat{T2}$	$\left(\widehat{T2}-T1\right)/T0$
10	0.158	0.414	1.591	7.449
30		1.120	5.136	25.418
50		2.274	10.613	52.778

variables T0, T1, $\widehat{\mathrm{T2}}$, and $\left(\widehat{\mathrm{T2}}-\mathrm{T1}\right)/\mathrm{T0}$ for each of the observed dimensions $D=\{10,30,50\}$. This calculation is independent of the computing system and programming language in which the measured algorithm is implemented. According to Table 2, it is easy to conclude that more computational cost is required with the increasing number of dimensions for the benchmark functions.

4.2 Experimental conditions and parameters setting

For a fair comparison among all the algorithms, they are executed using the same maximum number of function evaluations ($EFs = D \times 10000$), where the D is the size of the dimension. All the algorithms are executed independently 51 times on each test function. The experiments are run in the Windows Server 2012 R2 under the hardware environment of Intel (R) Core (TM) i7-6700 CPU @ 3.40 GHz processor and 8.0 GB of RAM. The proposed algorithm is implemented using the C + + programming language. It is necessary to point out that a numerical analysis and processing library, ALGLIB (http://www.alglib.net/), is utilized to implement the process of LBFGS. For each comparison algorithm, the values of the control parameter are configured as recommended in the corresponding literature.

The value of the parameters has a significant impact on the performance of the jSO_CMA-ES_LBFGS. There are three crucial parameters: (1) stagnation threshold T; (2) memory size H; (3) the step length σ in the jSO_CMA-ES_LBFGS. The Taguchi method of design is adopted to calibrate the value of each parameter in jSO_CMA-ES_LBFGS. In the experiment, the proposed algorithm is calibrated using the 30-dimensional CEC2017 benchmark function [41]. The various values of T, H, and σ are listed in Table 3. The parameter combinations and average error values yielded by jSO_CMA-ES_LBFGS are listed in Table 4. According to Table 4, the significance rank of each parameter is listed in Table 5. Meanwhile, the tendency of the parameters is described in Fig. 5.

Table 3 The parameters for different levels

Parameters	Levels		
	1	2	3
H	15	10	5
T	5	10	20
σ	0.1	0.5	1

Table 4 Parameter combinations

No.	Parameter combinations			Average error values
	H	T	σ	
1	15	5	0.1	2.6527E + 02
2	15	10	0.5	2.6684E + 02
3	15	20	1	2.6665E + 02
4	10	5	0.5	2.5748E + 02
5	10	10	1	2.5933E + 02
6	10	20	0.1	2.5877E + 02
7	5	5	1	2.5195E + 02
8	5	10	0.1	**2.4776E + 02**
9	5	20	0.5	2.5290E + 02

Table 5 Response table for means

Levels	H	T	σ
1	2.6630E + 02	2.5820E + 02	**2.5730E + 02**
2	2.5850E + 02	**2.5800E + 02**	2.5910E + 02
3	**2.5090E + 02**	2.5940E + 02	2.5930E + 02
Delta	1.5400E + 01	1.5000E + 00	2.0000E + 00
Rank	1	3	2

The optimal level value of each parameter is marked in bold

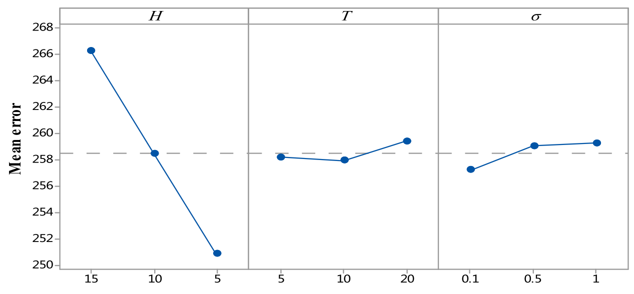

Fig. 5 Main effects plot of all parameters

From Table 5, it is observed that H is the most significant one among all the parameters, which implies that the memory size H is an important influential factor to jSO_CMA-ES_BFGS. The large H records more search history information. The external storage failure is caused by abnormal H with a too large or too small value. The σ ranks the second place, which illustrates that it is also an important factor in jSO_CMA-ES_LBFGS. A small σ would cause the population easily to fall into the local optimum. A large σ leads the algorithm to search in the global space and wastes a lot of function evaluations, similar to the random search. The stagnation threshold parameter T ranks third place. T is the best combination point of the global search operator and the local search operator to achieve a trade-off between exploration and exploitation.

According to the results of the Taguchi method, the parameters in jSO_CMA-ES_LBFGS are suggested as follows: $T = 10, H = 5, \sigma = 0.1$.

4.3 Selection of local search operators

The two local search algorithms, Solis&Wets algorithm (SW) and the Multi-Trajectory Local Search (Mtsls1) are tested in the paper. Although various adjustments in the parameter settings are combined to solve the problem in this paper, the essences of these comparison algorithms (e.g., the framework, encoding, operating mechanism) are not changed in the implementation process. According to the single factor principle, the parameters of jSO and CMA-ES remain unchanged. The parameter values used by SW and MTSLS1 are shown in Table 6. The experimental results of LBFGS, SW, and MTSLS1 are shown in Tables 7, 8, 9.

The SW algorithm belongs to the estimation of distribution algorithm (EDA), the number of parameters of the SW algorithm is more than that of MTSLS1 and LBFGS. There is no prior knowledge to find the best combination of parameters of the SW algorithm. The Mtsls1 algorithm searches from one dimension to another without considering the dependencies between variables. LBFGS belongs to the iterative gradient decrement algorithm. LBFGS is known for the fast convergence for solving convex optimization problems and achieves higher accuracy with a fewer number of function evaluations.

From Table 7, the LBFGS is significantly better than SW and Mtsls1 on 29 benchmark functions with $\alpha = 0.1$ and $\alpha = 0.05$ for 30 dimensions. Therefore, the LBFGS is employed as a local search operator in the jSO_CMA-ES_LBFGS rather than the SW or MTSLS1.

4.4 The effect of ensemble operators

As shown in Figs. 6 and 7. If there is no LBFGS operator in jSO_CMA-ES_LBFGS, jSO_CMA-ES_LBFGS is degraded to jSO_CMA-ES. The diversity of the algorithm in the process of solving is increased effectively by the gray points in Fig. 6 (the solutions generated by CMA-ES). The mean values of jSO, jSO_CMA-ES, and jSO_LBFGS on 30 dimensions and 50 dimensions are showed in Table 10. From Table 11, it is found that jSO_CMA-ES is significantly better than jSO on 29 benchmark functions with $\alpha = 0.1$ for 30 dimensions. Therefore, CMA-ES is embedded in jSO and has at least no side effects on the performance of jSO. According to the literature [42–44], CMA-ES has the characteristics of de-randomization and rotational invariance. New search points are sampled by normal distribution as perturbations of the mean (redpoint). The standard CMA-ES belongs to the estimation of the distribution algorithm, which has a strong global search ability and insufficient local searchability. It is not suitable for solving high-dimensional optimization problems.

Table 6 The parameter values of SW and MTSLS1

Algorithms	The parameter values
SW	Initial step size:$b_0 = (0, 0, \cdots 0)$
	Initial standard deviation $\sigma_0 = (1, 1, \cdots 1)$
	Local search parameter:$\rho = 1$
	The maximum number of failures for decreasing ρ:$s_{ex} = 5$
	The maximum number of Successes for increasing ρ:$f_{ct} = 3$
	The exponential factor for increasing ρ:$ex = 2$
	The contraction factor for decreasing ρ: $ct = 0.5$
	The maximum number of local search iterations:$K = 10$
MTSLS1	Search range:$SR = 1$
	The contraction factor for decreasing SR:$\rho = 0.1$
	The minimum threshold for the SR:$\varepsilon = 10^\wedge - 8$
	Reward factor: $BONUS1 = 10, BONUS2 = 1$
	The maximum number of local search iterations:$K = 20$

Table 7 Results of the multiple-problem Wilcoxon's test

D	jSO_CMA-ES_LBFGS *vs*	*R*+	*R*−	*Z*	*p*-value	α = 0.1	α = 0.05
10	jSO_CMA-ES_SW	118.50	134.50	− 0.26	0.795	No	No
	jSO_CMA-ES_MTSLS1	70.50	139.50	− 1.29	0.198	No	No
30	jSO_CMA-ES_SW	37.00	263.00	− 3.23	0.001	**Yes**	**Yes**
	jSO_CMA-ES_MTSLS1	31.00	245.00	− 3.25	0.001	**Yes**	**Yes**

The best mean value for each function is highlighted in boldface

Table 8 The mean error of different local search algorithm combinations on 10 dimensions

Func	jSO_CMA-ES_LBFGS Mean $_{\text{Std}}$	jSO_CMA-ES_SW Mean $_{\text{Std}}$	jSO_CMA-ES_MTSLS1 Mean $_{\text{Std}}$
f_1	**0.00E + 00**$_{0.00E+00}$	**0.00E + 00**$_{0.00E+00}$	**0.00E + 00**$_{0.00E+00}$
f_3	**0.00E + 00**$_{0.00E+00}$	**0.00E + 00**$_{0.00E+00}$	**0.00E + 00**$_{0.00E+00}$
f_4	**0.00E + 00**$_{0.00E+00}$	**0.00E + 00**$_{0.00E+00}$	**0.00E + 00**$_{0.00E+00}$
f_5	1.76E + 00$_{8.38E-01}$	**1.68E + 00**$_{7.49E-01}$	1.93E + 00$_{7.21E-01}$
f_6	**0.00E + 00**$_{0.00E+00}$	**0.00E + 00**$_{0.00E+00}$	**0.00E + 00**$_{0.00E+00}$
f_7	1.21E + 01$_{6.05E-01}$	1.20E + 01$_{6.24E-01}$	**1.20E + 01**$_{7.01E-01}$
f_8	2.03E + 00$_{9.21E-01}$	**1.90E + 00**$_{7.68E-01}$	2.09E + 00$_{7.44E-01}$
f_9	**0.00E + 00**$_{0.00E+00}$	**0.00E + 00**$_{0.00E+00}$	**0.00E + 00**$_{0.00E+00}$
f_{10}	**5.50E + 01**$_{5.18E+01}$	7.29E + 01$_{7.85E+01}$	8.03E + 01$_{8.29E+01}$
f_{11}	**0.00E + 00**$_{0.00E+00}$	1.84E-07$_{9.64E-07}$	7.21E-07$_{1.53E-06}$
f_{12}	**5.50E-01**$_{2.32E+01}$	1.46E + 01$_{3.84E+01}$	1.50E + 01$_{3.89E+01}$
f_{13}	3.48E + 00$_{2.40E+00}$	**2.96E + 00**$_{2.40E+00}$	3.90E + 00$_{2.13E+00}$
f_{14}	**2.31E-02**$_{2.96E-01}$	5.85E-02$_{2.34E-01}$	1.07E-01$_{3.01E-01}$
f_{15}	**2.98E-01**$_{2.00E-01}$	3.29E-01$_{1.94E-01}$	3.22E-01$_{2.12E-01}$
f_{16}	**5.17E-01**$_{2.84E-01}$	6.60E-01$_{2.34E-01}$	6.16E-01$_{1.87E-01}$
f_{17}	**4.31E-01**$_{4.11E-01}$	4.54E-01$_{3.23E-01}$	5.00E-01$_{3.68E-01}$
f_{18}	**2.45E-01**$_{2.16E-01}$	2.68E-01$_{1.99E-01}$	3.46E-01$_{1.85E-01}$
f_{19}	**9.61E-03**$_{2.08E-02}$	1.08E-02$_{2.22E-02}$	1.01E-02$_{1.50E-02}$
f_{20}	7.84E-01$_{1.29E-01}$	**2.70E-01**$_{2.01E-01}$	3.19E-01$_{1.50E-01}$
f_{21}	1.32E + 02$_{4.77E+01}$	**1.22E + 02**$_{4.23E+01}$	1.30E + 02$_{4.69E+01}$
f_{22}	**1.00E + 02**$_{3.66E-13}$	**1.00E + 02**$_{1.98E-13}$	**1.00E + 02**$_{2.38E-13}$
f_{23}	3.01E + 02$_{1.47E+00}$	**2.95E + 02**$_{4.17E+01}$	3.01E + 02$_{1.37E+00}$
f_{24}	2.88E + 02$_{9.42E+01}$	2.56E + 02$_{1.06E+02}$	**2.50E + 02**$_{1.11E+02}$
f_{25}	4.12E + 02$_{1.92E+01}$	**4.11E + 02**$_{2.08E+01}$	4.12E + 02$_{2.11E+01}$
f_{26}	**3.00E + 02**$_{0.00E+00}$	**3.00E + 02**$_{0.00E+00}$	**3.00E + 02**$_{0.00E+00}$
f_{27}	**3.88E + 02**$_{1.67E-01}$	3.89E + 02$_{1.31E+00}$	3.89E + 02$_{1.35E+00}$
f_{28}	3.15E + 02$_{1.99E+01}$	3.22E + 02$_{6.95E+01}$	**3.12E + 02**$_{5.72E+01}$
f_{29}	**2.35E + 02**$_{3.21E+00}$	2.37E + 02$_{4.62E+00}$	2.39E + 02$_{4.36E+00}$
f_{30}	**3.94E + 02**$_{4.54E-01}$	3.96E + 02$_{9.34E+00}$	3.96E + 02$_{9.35E+00}$

The best mean value for each function is highlighted in boldface

Here, CMA-ES only as an agent provides a relatively reliable initial solution for LBFGS. As shown in Table 11, a significant difference between jSO_CMA-ES and jSO with $\alpha = 0.1$ is not observed for 50-dimensional optimization problems, which demonstrates the rationality of the hypothesis proposed in this paper. As shown in Fig. 7. If there is no CMA-ES operator in jSO_CMA-ES_LBFGS, jSO_CMA-ES_LBFGS is degraded to jSO_LBFGS. It makes the search area limited to a locally optimal solution to fall into stagnation. As shown in Table 11, a significant difference between jSO_LBFGS and jSO with $\alpha = 0.1$ is also not observed. Therefore, the disturbance strength of jSO_LBFGS is insufficient. However, as shown in Table 12, jSO_CMA-ES_LBFGS is significantly better than jSO with $\alpha = 0.1$ and $\alpha = 0.05$ for 30 and 50 dimensions.

Table 9 The mean error of different local search algorithm combinations on 30 dimensions

Func	jSO_CMA-ES_LBFGS Mean_{Std}	jSO_CMA-ES_SW Mean_{Std}	jSO_CMA-ES_MTSLS1 Mean_{Std}
f_1	$\mathbf{0.00E+00}_{0.00E+00}$	$\mathbf{0.00E+00}_{0.00E+00}$	$4.21E\text{-}04_{1.17E\text{-}03}$
f_3	$\mathbf{0.00E+00}_{0.00E+00}$	$\mathbf{0.00E+00}_{0.00E+00}$	$1.56E\text{-}04_{5.82E\text{-}04}$
f_4	$\mathbf{1.44E+01}_{2.26E+01}$	$3.66E+01_{8.17E+00}$	$5.60E+01_{1.21E+01}$
f_5	$8.72E+00_{1.92E+00}$	$9.03E+00_{1.83E+00}$	$\mathbf{8.50E+00}_{1.20E+00}$
f_6	$8.23E\text{-}06_{2.92E\text{-}05}$	$5.40E\text{-}08_{1.64E\text{-}07}$	$\mathbf{2.05E\text{-}08}_{4.89E\text{-}08}$
f_7	$3.86E+01_{1.71E+00}$	$\mathbf{3.77E+01}_{1.38E+00}$	$3.81E+01_{7.18E\text{-}01}$
f_8	$9.02E+00_{1.98E+00}$	$9.31E+00_{1.75E+00}$	$\mathbf{8.90E+00}_{1.37E+00}$
f_9	$\mathbf{0.00E+00}_{0.00E+00}$	$\mathbf{0.00E+00}_{0.00E+00}$	$\mathbf{0.00E+00}_{0.00E+00}$
f_{10}	$\mathbf{1.53E+03}_{2.29E+02}$	$1.67E+03_{2.66E+02}$	$1.63E+03_{1.91E+02}$
f_{11}	$\mathbf{2.66E+00}_{2.45E+00}$	$7.96E+00_{1.58E+01}$	$7.31E+00_{1.25E+01}$
f_{12}	$\mathbf{1.61E+02}_{1.02E+02}$	$3.80E+02_{1.89E+02}$	$4.44E+02_{2.34E+02}$
f_{13}	$\mathbf{1.75E+01}_{5.67E+00}$	$1.90E+01_{5.79E+00}$	$2.08E+01_{4.19E+00}$
f_{14}	$\mathbf{2.16E+01}_{2.67E+00}$	$2.18E+01_{1.12E+00}$	$2.17E+01_{1.13E+00}$
f_{15}	$\mathbf{1.72E+00}_{1.29E+00}$	$2.13E+00_{1.59E+00}$	$2.20E+00_{1.55E+00}$
f_{16}	$\mathbf{5.23E+01}_{7.44E+01}$	$1.14E+02_{9.33E+01}$	$1.10E+02_{9.26E+01}$
f_{17}	$\mathbf{3.30E+01}_{7.79E+00}$	$3.67E+01_{7.11E+00}$	$3.85E+01_{6.20E+00}$
f_{18}	$\mathbf{2.07E+01}_{3.16E\text{-}01}$	$2.10E+01_{5.98E\text{-}01}$	$2.10E+01_{5.64E\text{-}01}$
f_{19}	$\mathbf{4.18E+00}_{1.66E+00}$	$5.71E+00_{2.16E+00}$	$5.85E+00_{2.26E+00}$
f_{20}	$\mathbf{2.86E+01}_{7.63E+00}$	$3.30E+01_{6.87E+00}$	$3.71E+01_{7.67E+00}$
f_{21}	$2.09E+02_{2.09E+00}$	$\mathbf{2.08E+02}_{1.93E+00}$	$2.09E+02_{1.66E+00}$
f_{22}	$\mathbf{1.00E+02}_{5.55E\text{-}12}$	$\mathbf{1.00E+02}_{1.24E\text{-}12}$	$\mathbf{1.00E+02}_{8.14E\text{-}09}$
f_{23}	$3.50E+02_{2.79E+00}$	$\mathbf{3.49E+02}_{4.53E+00}$	$3.50E+02_{4.24E+00}$
f_{24}	$4.26E+02_{2.05E+00}$	$\mathbf{4.24E+02}_{2.58E+00}$	$4.25E+02_{2.31E+00}$
f_{25}	$\mathbf{3.80E+02}_{1.35E+00}$	$3.86E+02_{1.07E\text{-}01}$	$3.87E+02_{1.68E\text{-}02}$
f_{26}	$\mathbf{9.19E+02}_{3.47E+01}$	$9.44E+02_{4.28E+01}$	$9.43E+02_{3.61E+01}$
f_{27}	$\mathbf{4.94E+02}_{9.09E+00}$	$4.97E+02_{6.64E+00}$	$5.00E+02_{7.50E+00}$
f_{28}	$3.06E+02_{2.57E+01}$	$3.11E+02_{3.33E+01}$	$\mathbf{3.06E+02}_{2.48E+01}$
f_{29}	$\mathbf{4.33E+02}_{1.45E+01}$	$4.38E+02_{8.43E+00}$	$4.43E+02_{6.21E+00}$
f_{30}	$\mathbf{1.97E+03}_{1.99E+01}$	$1.97E+03_{3.41E+01}$	$1.97E+03_{1.55E+01}$

The best mean value for each function is highlighted in boldface

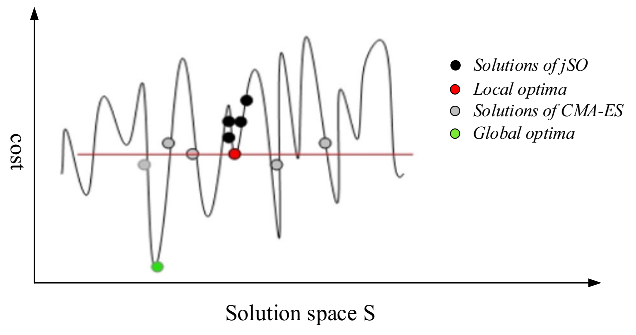

Fig. 6 The operating mechanism of jSO_CMA-ES

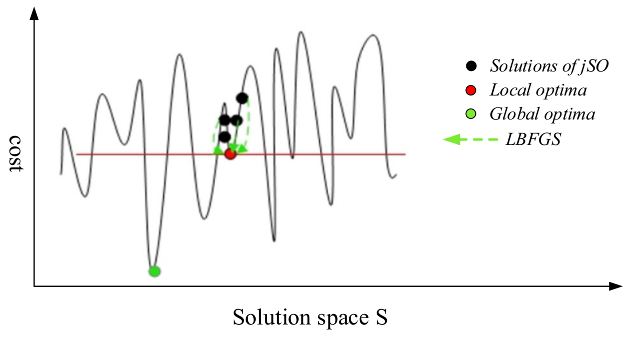

Fig. 7 The operating mechanism of jSO_LBFGS

The mean plot with a 95% confidence interval for the different operators is shown in Fig. 8. The abscissa represents three different kinds of functions including simple multimodal functions ($f_4 - f_{10}$), hybrid functions ($f_{11} - f_{20}$), and composition functions ($f_{21} - f_{30}$) [19]. The horizontal axis is the different operators. The vertical axis is the normalized value of 51 independent tests by Eq. (45). To make the effect clear, the values of 51 independent tests were normalized as

Table 10 The mean values of jSO, jSO_CMA-ES, and jSO_LBFGS

Func.	30D			50D		
	jSO	jSO_CMA-ES	jSO-LBFGS	jSO	jSO_CMA-ES	jSO-LBFGS
f_1	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
f_3	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
f_4	5.87E + 01	1.38E + 01	**0.00E + 00**	5.62E + 01	3.51E + 01	**3.13E-01**
f_5	8.84E + 00	8.94E + 00	**8.24E + 00**	1.64E + 01	**1.61E + 01**	1.71E + 01
f_6	3.09E-08	**9.39E-09**	9.39E-05	1.09E-06	**4.33E-07**	7.00E-03
f_7	3.90E + 01	**3.85E + 01**	3.87E + 01	**6.65E + 01**	6.70E + 01	6.71E + 01
f_8	9.47E + 00	9.07E + 00	**8.93E + 00**	**1.70E + 01**	1.80E + 01	1.72E + 01
f_9	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
f_{10}	**1.53E + 03**	1.56E + 03	1.54E + 03	**3.14E + 03**	3.15E + 03	3.19E + 03
f_{11}	7.57E + 00	3.79E + 00	**2.52E + 00**	2.79E + 01	2.27E + 01	**1.30E + 01**
f_{12}	1.83E + 02	**1.73E + 02**	1.80E + 02	1.68E + 03	**1.67E + 03**	1.68E + 03
f_{13}	**1.54E + 01**	**1.54E + 01**	1.59E + 01	**3.06E + 01**	3.47E + 01	3.25E + 01
f_{14}	**2.12E + 01**	2.19E + 01	2.13E + 01	2.50E + 01	2.49E + 01	**2.48E + 01**
f_{15}	**1.17E + 00**	1.29E + 00	1.75E + 00	2.39E + 01	**2.30E + 01**	2.40E + 01
f_{16}	9.43E + 01	7.59E + 01	**5.21E + 01**	4.51E + 02	4.39E + 02	**4.17E + 02**
f_{17}	3.37E + 01	3.28E + 01	**3.24E + 01**	2.83E + 02	**2.75E + 02**	3.00E + 02
f_{18}	2.09E + 01	**2.00E + 01**	2.04E + 01	**2.43E + 01**	2.46E + 01	2.46E + 01
f_{19}	4.44E + 00	**3.82E + 00**	4.66E + 00	1.41E + 01	**1.35E + 01**	1.37E + 01
f_{20}	**2.79E + 01**	2.90E + 01	2.93E + 01	1.40E + 02	**1.38E + 02**	1.60E + 02
f_{21}	**2.09E + 02**	2.10E + 02	2.10E + 02	2.19E + 02	**2.18E + 02**	**2.18E + 02**
f_{22}	**1.00E + 02**	**1.00E + 02**	**1.00E + 02**	**1.49E + 03**	1.81E + 03	1.52E + 03
f_{23}	3.51E + 02	3.51E + 02	**3.50E + 02**	**4.30E + 02**	4.31E + 02	4.31E + 02
f_{24}	4.27E + 02	**4.26E + 02**	**4.26E + 02**	**5.07E + 02**	**5.07E + 02**	5.08E + 02
f_{25}	3.87E + 02	3.82E + 02	**3.80E + 02**	4.81E + 02	**4.79E + 02**	**4.79E + 02**
f_{26}	9.38E + 02	9.33E + 02	**9.32E + 02**	**1.13E + 03**	1.15E + 03	1.15E + 03
f_{27}	4.95E + 02	4.93E + 02	**4.92E + 02**	5.11E + 02	**5.10E + 02**	5.11E + 02
f_{28}	**3.04E + 02**	3.09E + 02	3.06E + 02	4.60E + 02	4.44E + 02	**4.39E + 02**
f_{29}	4.33E + 02	**4.31E + 02**	4.34E + 02	**3.63E + 02**	3.65E + 02	**3.63E + 02**
f_{30}	**1.97E + 03**	**1.97E + 03**	**1.97E + 03**	6.01E + 05	6.07E + 05	**1.74E + 03**

The best value of each function is highlighted in boldface

Table 11 The results of the Wilcoxon's test for jSO, jSO_CMA-ES, and jSO_LBFGS

D	jSO *vs*	$R+$	$R-$	Z	$p-$ value	$\alpha = 0.1$
30	jSO_CMA-ES	180.50	72.50	− 1.754	0.079	**Yes**
	jSO_LBFGS	204.50	95.50	− 1.558	0.119	No
50	jSO_CMA-ES	180.00	145.00	− 0.471	0.637	No
	jSO_LBFGS	129.50	146.50	− 0.259	0.796	No

Table 12 The results of the Wilcoxon's test for jSO_CMA-ES_LBFGS, and jSO

D	jSO_CMA-ES_LBFGS *vs*	$R+$	$R-$	Z	p. -value	$\alpha = 0.1$	$\alpha = 0.05$
30	jSO	53.00	178.00	− 2.173	0.030	**Yes**	**Yes**
50	jSO	65.50	187.50	− 1.981	0.048	**Yes**	**Yes**

$$\text{Normalized Values} = \log_{10}(\text{Mean of51 independent tests}) \quad (45)$$

The points in the figure below reflect the excellent performance of the operators, and the horizontal lines on both sides of each point represent the stability of the

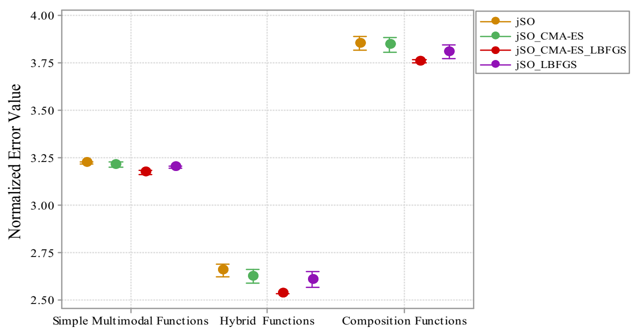

Fig. 8 Mean plot with a 95% confidence interval for the different operators

operators. The results show that each strategy contributes to the jSO_CMA-ES_LBFGS. In particular, the results have a significant improvement compared with the original jSO in hybrid functions and composition functions. Therefore, jSO, CMA-ES, and LBFGS are inseparable and indispensable organic whole, which achieves the effect of "$1 + 1 + 1 > 3$" instead of "$1 + 1 > 2$".

4.5 Comparison of jSO_CMA-ES_LBFGS with certain state-of-the-art algorithms

In this section, the proposed jSO_CMA-ES_LBFGS is compared with eleven state-of-the-art algorithms, including jSO [19], LSHADE_cnEpSin [20], AMECoDEs [45], ELSHADE [46], EBLSHADE [46], EBOwithCMAR [47], LSHADE_SPACMA [26], LSHADE-EpSin [48], CMA-ES [44], EWWO [49], and OLPSO [50]. The jSO, LSHADE_cnEpSin, AMECoDEs, ELSHADE, EBLSHADE, LSHADE_SPACMA, and LSHADE-EpSin are typical variants of DE. Significantly, jSO, LSHADE-cnEpSin, and LSHADE_SPACMA are on the second, third, and fourth place on CEC2017 competition on single objective real-parameter optimization. The EBOwithCMAR is in the first place on CEC2017. The EWWO is an improved water wave optimization algorithm enhanced by CMA-ES. The value of the control parameter of each compared algorithm is set to the value recommended in the original paper. The simulation experiment is carried out to evaluate the performance of the proposed jSO_CMA-ES_LBFGS on the CEC2017 test suite. All algorithms are executed independently 51 times on each test function, and the mean value and standard deviation (std) metrics are calculated.

The parameters of the compared algorithms use the same settings as in their original papers, which are listed in Table 13. For jSO_CMA-ES_LBFGS, the parameters used in these comparison experiments are set experimentally. Detailed parameter analysis about the jSO_CMA-ES_LBFGS is given in Sect. 4.2.

Meanwhile, the jSO_CMA-ES_LBFGS and other comparison algorithms are compared on the convergence rate. The convergence plots of the $f_1, f_4, f_7, f_{11}, f_{12}, f_{26}$, and f_{28} which include the unimodal, simple multimodal, hybrid, and composition functions are described in Fig. 9. The reason why these functions are selected for analysis is that they are the best reflection of the performance and operating mechanism of jSO_CMA-ES_LBFGS on all test functions. The mean value of jSO_CMA-ES_LBFGS and jSO are listed in Table 14.

As shown in Fig. 9, although the proposed algorithm does not converge at the fastest speed, it maintains a continuous downward trend and obtains a higher precision solution than that of the comparison algorithm, which is attributed to the role of perturbation operator and local search operator. In addition, a distinct separation between jSO and jSO_CMA-ES_LBFGS on the convergence rate for the late stage of population evolution is obvious from Fig. 9, which implicitly indicates that the operating mechanism of the proposed algorithm is effective.

The box plots for f_1, f_4, f_{11}, and f_{28} are shown in Fig. 10 to further illustrate the stability and performance of the proposed algorithm. Box plots use the data of five statistics: minimum, first quartile, median, and the third quartile and the maximum value to describe a method of data, it can also roughly see whether data have symmetry, and the distribution of the dispersed degree of information, such as special, can be used for the comparison of several samples. The sides of the two sides of boxes will correspond to the top and bottom quartile of the data batch. The red line in the boxes is the position of the median value. An extension line is created between the top quartile and the maximum, which is called "whicker". Similarly, build an extension line between the bottom quartile and the minimum. The points outside the extension line represent outliers. Therefore, the narrower the box and the fewer outliers mean that the data tend to be stable. On the other hand, the ordinate of the box plots represents the normalized error value, so the closer the box is to the bottom, the better the performance of the algorithm for the considered minimization problems. The stability of the algorithm is indicated from the box plots of $f_1(D = 30)$ and $f_{28}(D = 50)$ from Fig. 9. Although the narrowest box is not shown from the two box plots of $f_4(D = 30)$ and $f_{11}(D = 30)$, the box position of the proposed algorithm is the closest to the bottom, so the performance of the proposed algorithm is better than other comparison algorithms.

To testify the performance of the jSO_CMA-ES_LBFGS, Wilcoxon's test [51] is performed to check the behaviors of the six algorithms which are introduced as compared algorithms. The statistical analysis results are summarized in Table 15, considering jSO_CMA-ES_LBFGS as a control algorithm. The "yes" in bold

Table 13 Parameter settings

Algorithms	Authors	Parameter settings
jSO	Brest et al. [19]	$N^{init} = 25logD \cdot \sqrt{D}, H = 5,$ $\mu F = 0.5, \mu Cr = 0.8, N^{min} = 4$
LSHADE-cnEpsin	Awad et al. [20]	$N^{init} = 18 \cdot D, H = 5, \mu F = 0.5,$ $\mu Cr = 0.5, N^{min} = 4, freq = 0.5$
AMECoDEs	Cui et al. [45]	$N = 200, p = 0.1, c = 0.1, \varepsilon = 0.001$
ELSHADE	Mohamed et al. [46]	$N^{init} = 18 \cdot D, H = 5, \mu F = 0.5,$ $\mu Cr = 0.5, N^{min} = 4, \mu FCP = 0.5$
EBLSHADE	Mohamed et al. [46]	$N^{init} = 18 \cdot D, H = 5, \mu F = 0.5,$ $\mu Cr = 0.5, N^{min} = 4, \Delta_m \in [0.2, 0.8]$
EBOwithCMAR	Kumar et al. [47]	$PS_{1,max} = 18 \cdot D, PS_{1,min} = 4,$ $PS_{2,max} = 46.8D, PS_{2,min} = 10,$ $H = 6, PS_3 = 4 + (3\log(D)),$ $\sigma = 0.3, prob_{ls} = 0.1$
LSHADE-EpSin	Awad, et al. [48]	$N^{init} = 18 \cdot D, H = 5, \mu F = 0.5,$ $\mu Cr = 0.5, N^{min} = 4, \mu freq = 0.5,$ $freq = 0.5$
CMA-ES	Hansen et al. [44]	$N = D, x_{mean} = rand(n, 1), \sigma = 0.5,$ $\lambda = 4 + floor(3\log(N)),$ $\mu = ceil(\frac{\lambda-1}{2}), cc = \left(\frac{4}{N+2}\right),$ $cs = \left(\frac{\mu+2}{\mu+N+3}\right), eigeneval = 0$
EWWO	Zhao, et al. [49]	$CrMean = 0.5, pc = 0.8, k_{max} = 12$
LSHADE_SPACMA	Hadi, et al. [26]	$N^{init} = 18 \cdot D, H = 5, \mu F = 0.5,$ $\mu Cr = 0.5, N^{min} = 4$
OLPSO	Zhan et al. [50]	$\omega = 0.9, c = 2.0, G = 5,$ $V_{MAXd} = 0.2 * Range$

means that jSO_CMA-ES_LBFGS is significantly better than the another algorithm. From Table 15, it is found that jSO_CMA-ES_LBFGS is significantly better than jSO on 29 benchmark functions with $\alpha = 0.1$ for 30, 50, 100 dimensions and significantly better than AMECoDEs with $\alpha = 0.05$ and $\alpha = 0.1$ for 30, 50, and 100 dimensions. The jSO_CMA-ES_LBFGS is significantly better than ELSHADE with $\alpha = 0.1$ for 50 and 100 dimensions, and significantly better than EBLSHADE with $\alpha = 0.1$ for 10 and 100 dimensions. Although the significant differences are not observed between jSO_CMA-ES_LBFGS and the comparison algorithms in a certain case, the value of $R-$ is better than the value of $R+$. Namely, the jSO_CMA-ES_LBFGS obtains better solutions than other comparison algorithms in the above cases.

Friedman's test [51] is carried out to further test the significant differences between jSO_CMA-ES_LBFGS and the eleven competitors. The results are listed in Table 16. As shown in Fig. 11, there is a significant difference between jSO_CMA-ES_LBFGS and AMECoDEs with $\alpha = 0.1$ and $\alpha = 0.05$ for 30 dimensions. As shown in Fig. 12, a significant difference among jSO_CMA-ES_LBFGS, AMECoDEs, and ELSHADE with $\alpha = 0.1$ and $\alpha = 0.05$ is observed on 50 dimensions. Although there is no significant difference between jSO_CMA-ES_LBFGS and other compared algorithms in other cases, the rank of jSO_CMA-ES_LBFGS is the smallest in all comparison algorithms.

In summary, the results of the statistical analysis imply that the performance of the proposed jSO_CMA-ES_LBFGS is significantly better than that of the other comparison algorithms for test problems with 30 and 50 dimensions, whereas with the increase in problem dimensions, the performance of jSO_CMA-ES_LBFGS and the comparison algorithms decreases. On the one hand, the growing functional dimensions make the number of decision variables expand dramatically. On the other hand, in the CEC2017 benchmark test suite, the number of evaluations specified for functions with different dimensions is limited. Especially in high-dimensional functions, the

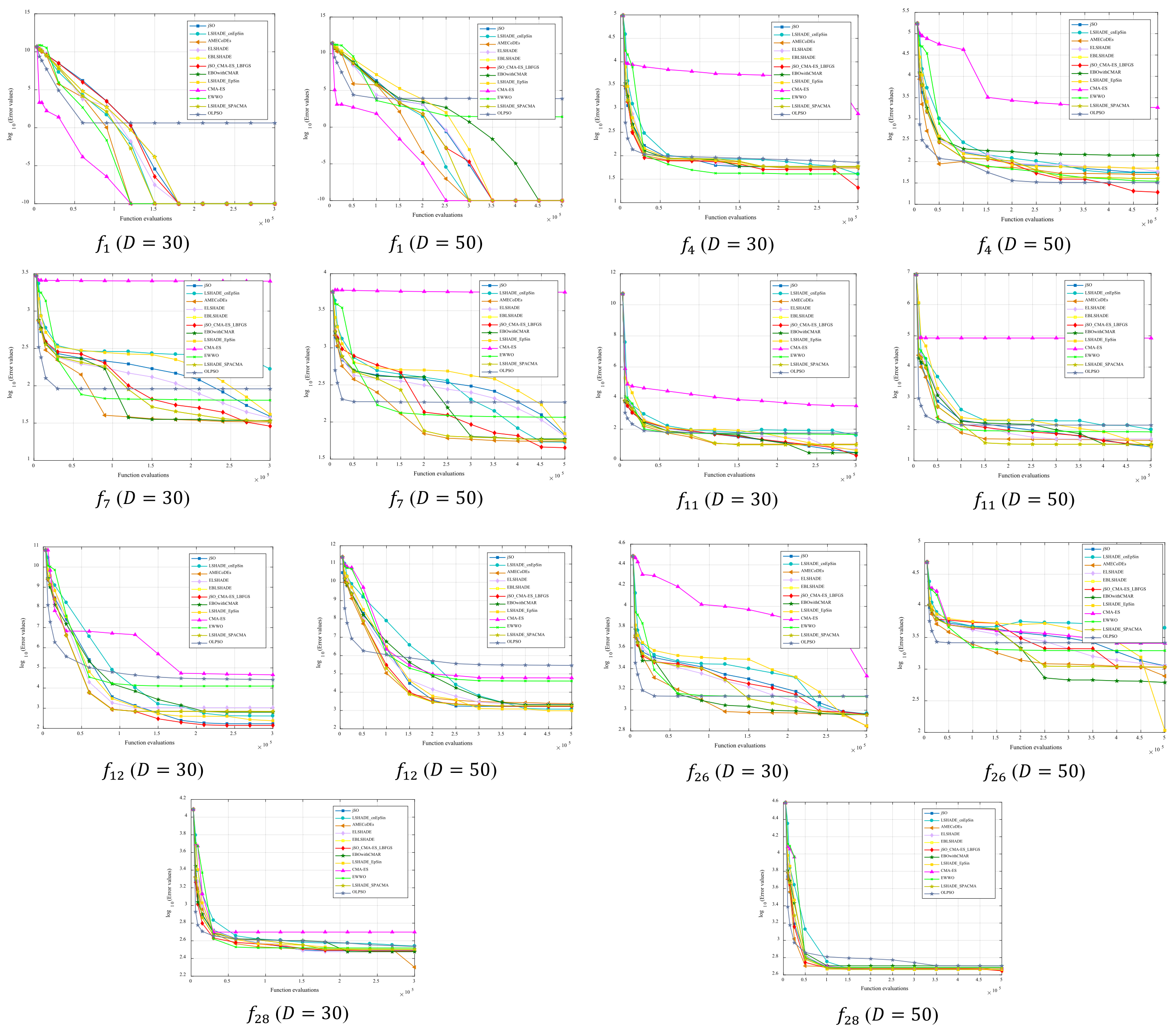

Fig. 9 Convergence plots of jSO_CMA-ES_LBFGS and other compared algorithms

algorithm cannot effectively find an approximate optimal solution within the limited number of evaluations. With the increase in function dimensions, the complexity of the problems to be solved increases sharply, which affects the performance of the algorithm.

5 Application to the engineering problems

The proposed algorithm is tested on two important engineering problems including continuous and discrete decision space. In addition, the performance of the jSO_CMA-ES_LBFGS is compared with canonical methods to analysis the advantages of the jSO_CMA-ES_LBFGS in addressing engineering optimization problems. For each engineering problem, the algorithms are run independently 31 times.

5.1 Gear train engineering design problem

The gear train engineering design problem is utilized to verify the performance of the jSO_CMA-ES_LBFGS in addressing engineering problems. Gear train design aims to minimize the gear ration of the gear train as shown in Fig. 13. There are four types of parameters in the gear train engineering design problem. The details of the mathematical model about this problem are described as follows [52].

Decision variable:

$$\vec{g} = [g_1, g_2, g_3, g_4] = [M_A, M_B, M_C, M_D] \tag{46}$$

Table 14 The mean value of jSO_CMA-ES_LBFGS and jSO

Func	30*D*		50*D*	
	jSO_CMA-ES_LBFGS	jSO	jSO_CMA-ES_LBFGS	jSO
f_1	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
f_3	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
f_4	**1.44E + 01**	5.87E + 01	**1.76E + 01**	5.62E + 01
f_5	**8.72E + 00**	8.84E + 00	1.73E + 01	**1.64E + 01**
f_6	8.23E-06	**3.09E-08**	1.06E-04	**1.09E-06**
f_7	**3.86E + 01**	3.90E + 01	**6.65E + 01**	**6.65E + 01**
f_8	**9.02E + 00**	9.47E + 00	1.71E + 01	**1.70E + 01**
f_9	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
f_{10}	**1.53E + 03**	**1.53E + 03**	**3.09E + 03**	3.14E + 03
f_{11}	**2.66E + 00**	7.57E + 00	**1.50E + 01**	2.79E + 01
f_{12}	**1.61E + 02**	1.83E + 02	**1.67E + 03**	1.68E + 03
f_{13}	1.75E + 01	**1.54E + 01**	4.50E + 01	**3.06E + 01**
f_{14}	2.16E + 01	**2.12E + 01**	**2.49E + 01**	2.50E + 01
f_{15}	1.72E + 00	**1.17E + 00**	2.43E + 01	**2.39E + 01**
f_{16}	**5.23E + 01**	9.43E + 01	**4.14E + 02**	4.51E + 02
f_{17}	**3.30E + 01**	3.37E + 01	3.03E + 02	**2.83E + 02**
f_{18}	**2.07E + 01**	2.09E + 01	**2.41E + 01**	2.43E + 01
f_{19}	**4.18E + 00**	4.44E + 00	**1.41E + 01**	1.41E + 01
f_{20}	2.86E + 01	**2.79E + 01**	**1.37E + 02**	1.40E + 02
f_{21}	2.09E + 02	**2.09E + 02**	**2.18E + 02**	2.19E + 02
f_{22}	**1.00E + 02**	**1.00E + 02**	**1.45E + 03**	1.49E + 03
f_{23}	**3.50E + 02**	3.51E + 02	4.30E + 02	**4.30E + 02**
f_{24}	**4.26E + 02**	4.27E + 02	5.08E + 02	**5.07E + 02**
f_{25}	**3.80E + 02**	3.87E + 02	**4.79E + 02**	4.81E + 02
f_{26}	**9.19E + 02**	9.38E + 02	1.14E + 03	**1.13E + 03**
f_{27}	**4.94E + 02**	4.95E + 02	**5.09E + 02**	5.11E + 02
f_{28}	3.06E + 02	**3.04E + 02**	**4.39E + 02**	4.60E + 02
f_{29}	4.33E + 02	**4.33E + 02**	3.63E + 02	**3.63E + 02**
f_{30}	1.97E + 03	**1.97E + 03**	**5.99E + 05**	6.01E + 05

The best value for each function is highlighted in boldface

Objective:

$$f(\vec{g}) = \left(\frac{1}{6.931} - \frac{g_2 g_3}{g_1 g_4}\right)^2 \tag{47}$$

Subject to:

$$12 \leq g_1, g_2, g_3, g_4 \leq 60 \tag{48}$$

The jSO_CMA-ES_LBFGS is run in 1000 fitness evaluations. The obtained statistical results for gear train engineering design problem are compared in Table 17. More details about Sandgren, GeneAS, and ABC are described in [52]. It is observed from Table 17 that the jSO_CMA-ES_LBFGS outperforms other algorithms.

5.2 The blocking flow shop scheduling problem

The blocking flow shop scheduling problem (BFSP) [53], which is one of the most important scheduling types, is widespread in modern industries. In the BFSP, there are no buffers between machines, and the job remains in the current machine until the next machine is available for processing. With the increase of the scheduling scale, the difficulty and computation time of solving the problem increased exponentially. Existing experiments and literature have shown that BFSP with more than two machines is a typical NP-hard problem [53].

The definitions of the variables and parameters mentioned in this section are recorded in Table 18. The details of the mathematical model of BFSP are described as follows.

Decision variable:

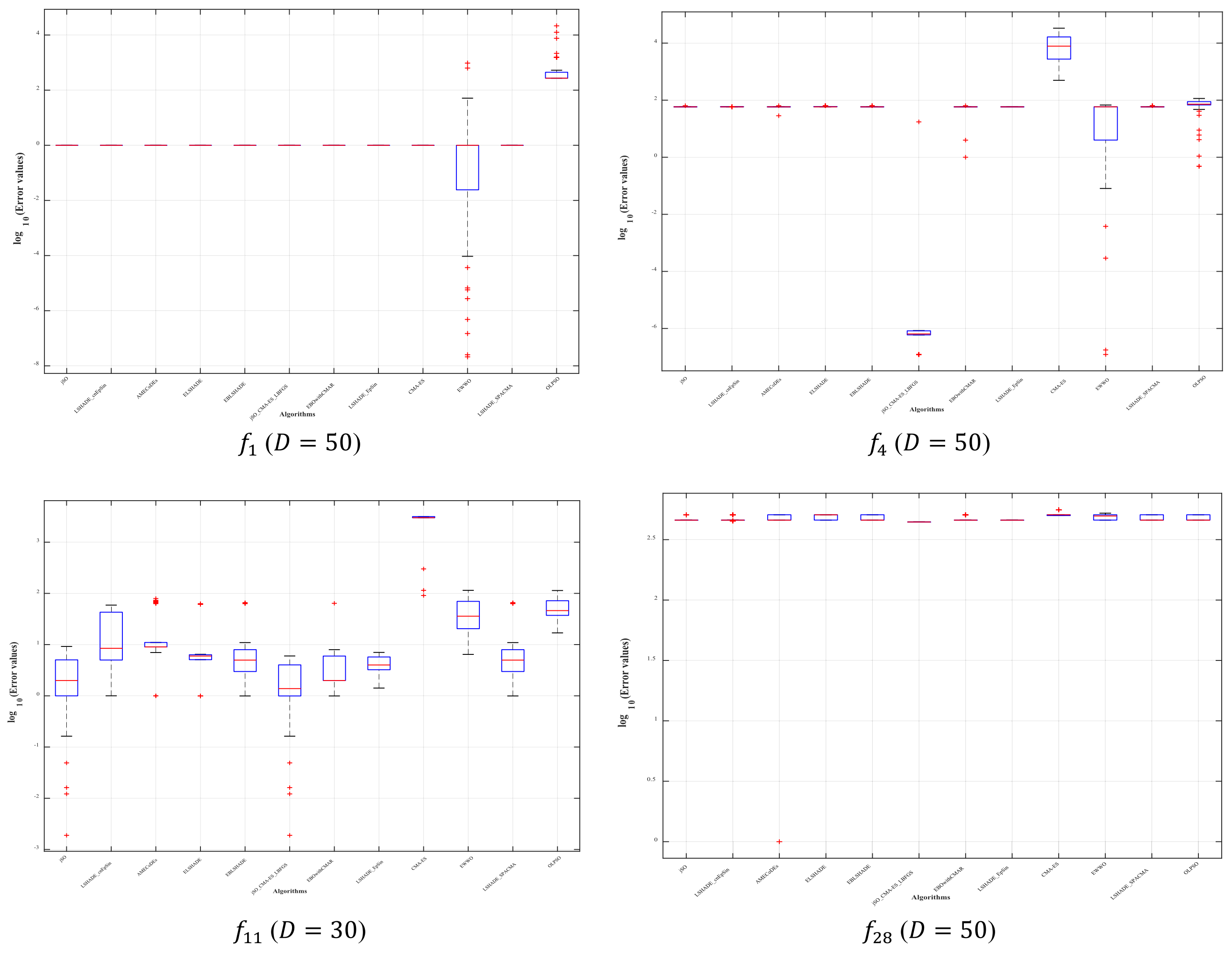

Fig. 10 Box plots of jSO_CMA-ES_LBFGS and other compared algorithms on some typical benchmark functions

$$x_{i,k} \in \{0, 1\}, (i, k = 1, 2, \ldots, n) \tag{49}$$

Objective:

$$\min C_{max} = \max_{k=1,2,\ldots,n} (D_{k,m}) \tag{50}$$

Subject to:

$$\sum_{k=1}^{n} x_{i,k} = 1, \quad i \in \{1, 2, \ldots, n\} \tag{51}$$

$$\sum_{i=1}^{n} x_{i,k} = 1, \quad k \in \{1, 2, \ldots, n\} \tag{52}$$

$$D_{1,0} \geq 0 \tag{53}$$

$$D_{k,0} \geq D_{k-1,1}, (k = 2, 3, \ldots, n) \tag{54}$$

$$D_{k,j} \geq D_{k,j-1} + \sum_{i=1}^{n} x_{i,k} \cdot P_{i,j}, \quad (k = 1, 2, \ldots, n), (j = 1, 2, \ldots, m) \tag{55}$$

$$D_{k,j} \geq D_{k-1,j+1}, \quad (k = 2, 3, \ldots, n), (j = 1, 2, \ldots, m - 1) \tag{56}$$

5.2.1 Mathematical model of BFSP

In this study, the objective is to minimize the makespan criterion. The makespan of the schedule π is $C_{max} = D_{n,m}$. The computation complexity of this task is $O(n*m)$.

$$[P_{i,j}]_{3\times3} = \begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,3} \\ P_{2,1} & P_{2,2} & P_{2,3} \\ P_{3,1} & P_{3,2} & P_{3,3} \end{bmatrix} = \begin{bmatrix} 3 & 2 & 1 \\ 1 & 3 & 2 \\ 2 & 1 & 3 \end{bmatrix} \tag{57}$$

Here, an example is presented to show how the decision variables reflect the solution by considering a problem with three jobs ($n = 3$) and three machines ($m = 3$). The processing times are given in Eq. (57). The scheduling Gantt is shown in Fig. 14. The makespan of schedule sets $\pi_1 = \{J_1, J_2, J_3\}$ in BFSP is 13.

Table 15 Results of the Wilcoxon's test for jSO_CMA-ES_LBFGS and other compared algorithms

D	jSO_CMA-ES_LBFGS *vs*	$R+$	$R-$	Z	$p-$ value	$\alpha = 0.1$	$\alpha = 0.05$
10	jSO	84.00	87.00	− 0.065	0.948	No	No
	LSHADE-cnEpSin	66.00	165.00	− 1.721	0.085	**Yes**	No
	AMECoDEs	131.50	168.50	− 0.529	0.597	No	No
	ELSHADE	94.00	159.00	− 1.055	0.291	No	No
	EBLSHADE	65.00	188.00	− 1.997	0.046	**Yes**	**Yes**
	EBOwithCMAR	85.00	86.00	− 0.064	0.886	No	No
	LSHADE-EpSin	59.00	172.00	− 1.827	0.049	**Yes**	**Yes**
	CMA-ES	15.00	228.00	− 2.916	0.000	**Yes**	**Yes**
	EWWO	18.00	225.00	− 2.855	0.000	**Yes**	**Yes**
	LSHADE_SPACMA	70.00	161.00	− 0.655	0.062	**Yes**	No
	OLPSO	25.00	218.00	− 2.641	0.001	**Yes**	**Yes**
30	jSO	53.00	178.00	− 2.173	0.030	**Yes**	No
	LSHADE-cnEpSin	59.00	241.00	− 2.600	0.009	**Yes**	**Yes**
	AMECoDEs	3.00	375.00	− 4.469	0.000	**Yes**	**Yes**
	ELSHADE	125.00	200.00	− 1.009	0.313	No	No
	EBLSHADE	160.00	165.00	− 0.067	0.946	No	No
	EBOwithCMAR	51.00	180.00	− 2.011	0.042	**Yes**	No
	LSHADE-EpSin	60.00	240.00	− 2.200	0.007	**Yes**	**Yes**
	CMA-ES	12.00	231.00	− 3.614	0.000	**Yes**	**Yes**
	EWWO	18.00	226.00	− 2.900	0.000	**Yes**	**Yes**
	LSHADE_SPACMA	78.00	222.00	− 0.732	0.059	**Yes**	No
	OLPSO	31.00	212.00	− 2.551	0.001	**Yes**	**Yes**
50	jSO	65.50	187.50	− 1.981	0.048	**Yes**	**Yes**
	LSHADE-cnEpSin	100.00	251.00	− 1.918	0.055	**Yes**	No
	AMECoDEs	15.00	420.00	− 4.379	0.000	**Yes**	**Yes**
	ELSHADE	78.00	273.00	− 2.476	0.013	**Yes**	**Yes**
	EBLSHADE	136.00	215.00	− 1.003	0.316	No	No
	EBOwithCMAR	63.00	190.00	− 2.090	0.490	No	No
	LSHADE-EpSin	71.00	229.00	− 1.716	0.041	**Yes**	**Yes**
	CMA-ES	11.00	232.00	− 4.255	0.000	**Yes**	**Yes**
	EWWO	19.00	224.00	− 3.915	0.000	**Yes**	**Yes**
	LSHADE_SPACMA	69.00	231.00	− 1.988	0.062	**Yes**	No
	OLPSO	36.00	217.00	− 2.499	0.001	**Yes**	**Yes**
100	jSO	117.00	289.00	− 1.959	0.050	**Yes**	No
	LSHADE-cnEpSin	276.50	129.50	− 1.674	0.094	**Yes**	No
	AMECoDEs	13.00	422.00	− 4.422	0.000	**Yes**	**Yes**
	ELSHADE	47.00	359.00	− 3.552	0.000	**Yes**	**Yes**
	EBLSHADE	120.00	286.00	− 1.890	0.059	**Yes**	No
	EBOwithCMAR	168.00	238.00	− 1.861	0.054	No	No
	LSHADE-EpSin	226.00	180.00	− 1.690	0.040	**Yes**	**Yes**
	CMA-ES	7.00	428.00	− 4.615	0.000	**Yes**	**Yes**
	EWWO	16.00	419.00	− 4.228	0.000	**Yes**	**Yes**
	LSHADE_SPACMA	129.00	277.00	− 1.855	0.072	**Yes**	No
	OLPSO	32.00	221.00	− 3.682	0.000	**Yes**	**Yes**

Table 16 Results achieved by Friedman Test

Algorithms	Mean Rank			
	10*D*	30*D*	50*D*	100*D*
jSO	3.090	3.190	2.840	2.740
LSHADE_cnEpSin	3.950	3.744	3.162	2.480
AMECoDEs	3.642	5.281	5.624	5.642
ELSHADE	3.592	3.241	3.953	4.342
EBLSHADE	3.673	2.810	2.880	3.344
EBOwithCMAR	**3.060**	2.751	2.600	2.571
LSHADE-EpSin	4.060	4.255	4.332	4.580
CMA-ES	5.012	6.206	6.712	7.710
EWWO	4.766	5.011	5.204	5.262
LSHADE_SPACMA	3.689	3.334	3.297	3.262
OLPSO	5.268	4.900	5.061	4.865
jSO_CMA-ES_LBFGS	3.072	**2.740**	**2.550**	**2.450**
Crit.Diff.$\alpha = 0.05$	1.2643			
Crit.Diff.$\alpha = 0.1$	1.1317			

The best mean rank for each dimension is highlighted in boldface

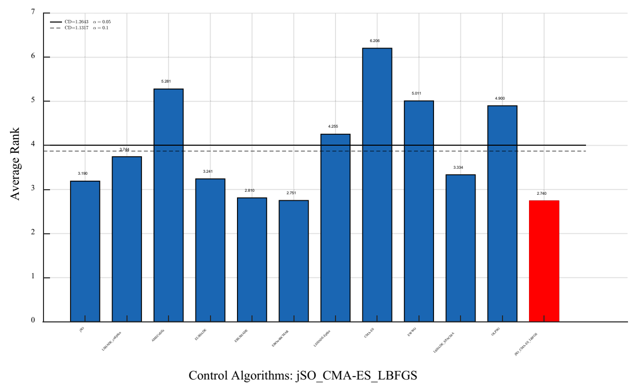

Fig. 11 Rankings for $D = 30$

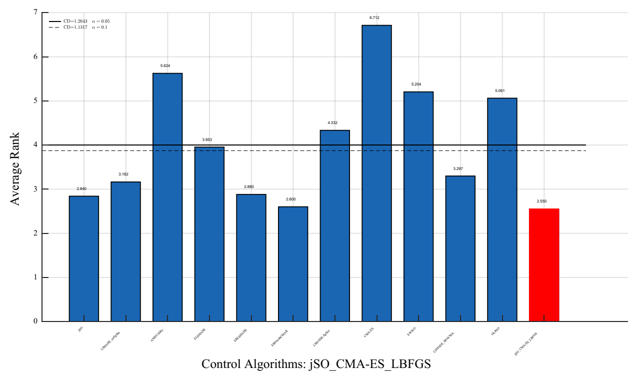

Fig. 12 Rankings for $D = 50$

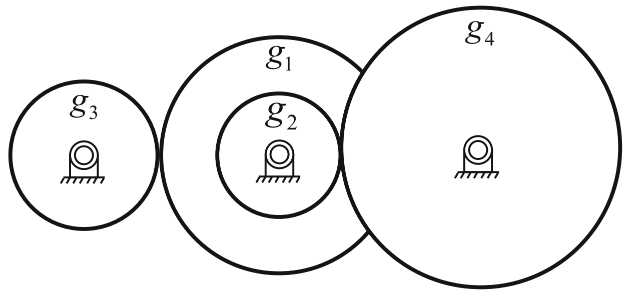

Fig. 13 Geat train design problem

5.2.2 Experimental settings and analysis

The traditional jSO algorithm and its variants cannot be directly utilized to solve the combinational optimization problem with discrete characteristics. Therefore, the coding scheme and decoding rule are utilized to help the algorithms work directly in the discrete domain by representing individuals as discrete job permutations. In this paper, the LOV rule is utilized to represented individuals as discrete job permutations. More details about LOV rule can be found in [54]. The average relative percentage deviation (ARPD) index is utilized to measure the results, and the calculate method is shown in Eq. (58), where R is the number of runs and C_i is the solution generated by a specific algorithm in the ith experiment for a given instance. C_{opt} is the minimum makespan found by all algorithms. The algorithm with minimum ARPD outperforms other algorithms.

$$ARPD = \frac{1}{R} \cdot \sum_{i=1}^{R} \frac{C_i - C_{opt}}{C_{opt}} \cdot 100\% \tag{58}$$

The well-known standard benchmark set of Taillard [55] is used for evaluating the performance of jSO_CMA-ES_LBFGS. This benchmark is composed of 120 different problem instances. The instances are categorized into 12 subsets of different combinations of n (number of jobs) and m (number of machines). These combinations range from 20 jobs and 5 machines up to 500 jobs and 20 machines. The results of jSO, OLPSO, EWWO, CMA-ES, EBLSHADE, LSHADE-cnEpsin, and jSO_CMA-ES_LBFGS are shown in Table 19. Each algorithm is run in $10\text{m} \cdot \text{n}$ milliseconds (ms). The parameters of the comparison algorithms are consistent with the previous ones.

The number of jobs and machines have a significant impact on all the compared algorithms from Table 19. The experiment results show the excellent performance of the jSO_CMA-ES_LBFGS for solving the BFSP. The effectiveness of jSO_CMA-ES_LBFGS compared with jSO is that the local search capability of proposed algorithm is enhanced via LBFGS. On the other hand, the LBFGS is embedded in CMA-ES to perturb the optimal candidates in

Table 17 The results of five algorithms

Algorithms	Sandgren	GeneAS	ABC	jSO	jSO_CMA-ES_LBFGS
g_1	6.00E + 01	5.00E + 01	4.90E + 01	5.11E + 01	6.00E + 01
g_2	4.50E + 01	3.00E + 01	1.60E + 01	1.90E + 01	4.33E + 01
g_3	2.20E + 01	1.40E + 01	1.90E + 01	2.75E + 01	1.20E + 01
g_4	1.80E + 01	1.70E + 01	4.30E + 01	7.10E + 01	6.00E + 01
$f(\vec{g})$	5.97E-01	1.22E-01	2.70E-12	7.00E-18	**2.92E-19**

Table 18 The definitions of the notations

Notations	Definition
n	Number of jobs
m	Number of machines
i	The index of the job, $i = 0, 1, 2, \cdots, n$. 0 is a virtual job
j	The index of the machine, $j = 1, 2, \cdots, m$
π	A feasible scheduling job sequence, $\pi = \{\pi(1), \pi(2), \ldots, \pi(n)\}$
$P_{i,j}$	Processing time of job i on machine j
k	The index of a certain job in the job sequence, $k \in \{1, 2, \ldots, n\}$
$D_{k,0}$	Start processing time of kth job on the first machine
$D_{k,j}$	Departure time of job kth job on machine j
$x_{i,k}$	The binary variable that takes value 1 if the kth job of π is i, and 0 otherwise
C_{max}	The maximum assembly completion time of the schedule π
$\{J_1, J_2, \ldots, J_n\}$	Job sequence
$\{M_1, M_2, \ldots, M_m\}$	Machine sequence

the solution space when the population is falling into stagnation. Therefore, jSO_CMA-ES_LBFGS is a competitive algorithm for solving BFSP.

6 Conclusions and future research

The paper presented a hybrid algorithm (jSO_CMA-ES_LBFGS) based on DE, CMA-ES, and LBFGS to solve the continuous optimization problems. The various experimental results imply the following conclusions: (1) LBFGS, as a local search operator, plays an important role in the proposed algorithm to enhance the local search capability of DE. (2) In the jSO_CMA-ES_LBFGS, a relatively reliable initial solution for the local search operator is generated by the CMA-ES, which is activated to perturb the optimal candidates in the solution space when the population is falling into stagnation. The LBFGS is embedded in CMA-ES as the local search strategy to obtain the potential local optimal solutions. (3) The search performance of jSO_CMA-ES_LBFGS is better than that of the other comparison algorithms on a certain confidence level. Further, jSO_CMA-ES_LBFGS is an effective

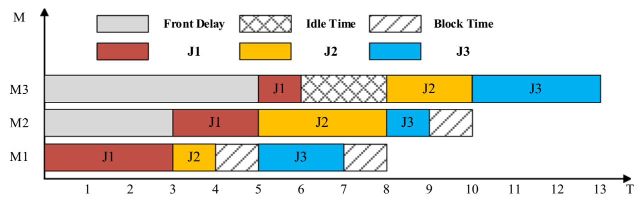

Fig. 14 Gantt chart for a solution to the example problem

algorithm to solve the continued optimization problems including the CEC2017 benchmarks and the gear train engineering design problem. On the other hand, the proposed jSO_CMA-ES_LBFGS is applied to solve BFSP effectively. The results for the benchmark set of Taillard demonstrate that the proposed algorithm is with good performance.

In future work, the search performance of jSO_CMA-ES_LBFGS is further improved, and the improved jSO_CMA-ES_LBFGS is applied to solve certain complex scheduling problems, e.g., the flow shop scheduling problems, the job shop scheduling problems, the lot-streaming flow shop scheduling problems. Moreover, DE is combined with other meta-heuristics for the multi-objective scheduling problems.

Table 19 The ARPD of all compared algorithms

$n * m$	jSO		OLPSO		EWWO		CMA-ES		EBLSHADE		LSHADE-cnEpsin		jSO_CMA-ES_LBFGS	
	ARPD	Std	ARPD	Std	ARPD	Std	ARPD	Std	ARPD	Std	ARPD	Std	ARPD	Std
20 × 5	0.046	0.099	3.890	0.081	1.514	0.089	1.101	0.133	0.340	0.142	0.160	0.073	**0.000**	0.042
20 × 10	0.086	0.110	3.610	0.063	1.621	0.113	1.051	0.135	0.341	0.144	0.215	0.089	**0.000**	0.032
20 × 20	0.015	0.082	1.470	0.064	0.968	0.101	0.807	0.125	0.125	0.125	0.080	0.070	**0.000**	0.022
50 × 5	0.309	0.121	7.299	0.115	4.936	0.085	1.056	0.118	1.383	0.121	1.040	0.119	**0.072**	0.077
50 × 10	0.305	0.121	4.452	0.111	4.578	0.070	0.917	0.123	1.087	0.129	0.905	0.087	**0.054**	0.075
50 × 20	0.269	0.080	3.346	0.074	4.021	0.061	0.856	0.084	0.815	0.080	0.795	0.041	**0.067**	0.034
100 × 5	0.140	0.013	7.304	0.115	4.846	0.765	0.340	0.210	0.855	0.526	0.879	0.236	**0.071**	0.083
100 × 10	**0.115**	0.018	4.934	0.084	4.820	0.533	0.253	0.183	0.925	0.390	0.738	0.372	0.120	0.075
100 × 20	0.098	0.021	3.551	0.167	4.570	0.419	0.202	0.137	0.854	0.356	0.618	0.587	**0.076**	0.022
200 × 10	0.161	0.489	5.778	0.449	3.651	0.493	0.621	0.258	0.711	0.884	0.705	0.484	**0.014**	0.252
200 × 20	0.134	0.542	3.971	0.431	4.106	0.637	0.545	0.129	0.675	0.741	0.481	0.141	**0.030**	0.599
500 × 20	0.104	0.424	5.316	0.468	2.458	0.782	0.520	0.131	0.312	0.744	0.905	0.812	**0.032**	0.472

The best ARPD for each instance is highlighted in boldface

Acknowledgements This work was financially supported by the National Natural Science Foundation of China under grant 62063021. It was also supported by Key talent project of Gansu Province (ZZ2021G50700016), the Key Research Programs of Science and Technology Commission Foundation of Gansu Province (21YF5WA086), Lanzhou Science Bureau project (2018-rc-98), Project of Zhejiang Natural Science Foundation (LGJ19E050001), respectively.

Declarations

Conflict of interest The authors declared that they have no conflict of interest.

References

1. Wang JJ, Wang L (2020) A knowledge-based cooperative algorithm for energy-efficient scheduling of distributed flow-shop. IEEE Trans Syst Man, Cybern Syst 50:1805–1819. https://doi.org/10.1109/TSMC.2017.2788879
2. Zhao F, He X, Wang L (2020) A two-stage cooperative evolutionary algorithm With problem-specific knowledge for energy-efficient scheduling of no-wait flow-shop problem. IEEE Trans Cybern. https://doi.org/10.1109/tcyb.2020.3025662
3. Zhao W, Wang L, Zhang Z (2020) Artificial ecosystem-based optimization: a novel nature-inspired meta-heuristic algorithm. Neural Comput Appl 32:9383–9425. https://doi.org/10.1007/s00521-019-04452-x
4. Hussain K, Salleh MNM, Cheng S, Shi Y (2019) On the exploration and exploitation in popular swarm-based metaheuristic algorithms. Neural Comput Appl 31:7665–7683. https://doi.org/10.1007/s00521-018-3592-0
5. Cao F (2020) PID controller optimized by genetic algorithm for direct-drive servo system. Neural Comput Appl 32:23–30. https://doi.org/10.1007/s00521-018-3739-z
6. Xu B, Cheng W, Qian F, Huang X (2019) Self-adaptive differential evolution with multiple strategies for dynamic optimization of chemical processes. Neural Comput Appl 31:2041–2061. https://doi.org/10.1007/s00521-018-03985-x
7. Wang K, Li X, Gao L et al (2021) A discrete artificial bee colony algorithm for multiobjective disassembly line balancing of end-of-life products. IEEE Trans Cybern. https://doi.org/10.1109/TCYB.2020.3042896
8. Hu Y, Zhang Y, Gong D (2020) Multiobjective particle swarm optimization for feature selection with fuzzy cost. IEEE Trans Cybern 51:874–888. https://doi.org/10.1109/tcyb.2020.3015756
9. Heidari AA, Aljarah I, Faris H et al (2020) An enhanced associative learning-based exploratory whale optimizer for global optimization. Neural Comput Appl 32:5185–5211. https://doi.org/10.1007/s00521-019-04015-0
10. Storn R, Price K (1997) Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11:341–359. https://doi.org/10.1023/A:1008202821328
11. Biswas PP, Suganthan PN, Wu G, Amaratunga GAJ (2019) Parameter estimation of solar cells using datasheet information with the application of an adaptive differential evolution algorithm. Renew Energy 132:425–438. https://doi.org/10.1016/j.renene.2018.07.152
12. Zhao F, Zhao L, Wang L, Song H (2020) An ensemble discrete differential evolution for the distributed blocking flowshop scheduling with minimizing makespan criterion. Expert Syst Appl. https://doi.org/10.1016/j.eswa.2020.113678
13. Li T, Pan Q, Gao L, Li P (2017) Differential evolution algorithm-based range image registration for free-form surface parts quality inspection. Swarm Evol Comput 36:106–123. https://doi.org/10.1016/j.swevo.2017.04.006
14. Wu G, Shen X, Li H et al (2018) Ensemble of differential evolution variants. Inf Sci (Ny) 423:172–186. https://doi.org/10.1016/j.ins.2017.09.053
15. Yang Z, Qiu H, Gao L et al (2020) Surrogate-assisted classification-collaboration differential evolution for expensive constrained optimization problems. Inf Sci (Ny) 508:50–63. https://doi.org/10.1016/j.ins.2019.08.054
16. Cai X, Gao L, Li X, Qiu H (2019) Surrogate-guided differential evolution algorithm for high dimensional expensive problems. Swarm Evol Comput 48:288–311. https://doi.org/10.1016/j.swevo.2019.04.009
17. Tian M, Gao X (2019) An improved differential evolution with information intercrossing and sharing mechanism for numerical

optimization. Swarm Evol Comput. https://doi.org/10.1016/j.swevo.2017.12.010
18. Noman N, Iba H (2008) Accelerating differential evolution using an adaptive local search. IEEE Trans Evol Comput 12:107–125. https://doi.org/10.1109/TEVC.2007.895272
19. Brest J, Maučec MS, Bošković B (2017) Single objective real-parameter optimization: Algorithm jSO. In: 2017 IEEE Congress on Evolutionary Computation, CEC 2017 - Proceedings. Institute of Electrical and Electronics Engineers Inc., pp 1311–1318
20. Awad NH, Ali MZ, Suganthan PN (2017) Ensemble sinusoidal differential covariance matrix adaptation with Euclidean neighborhood for solving CEC2017 benchmark problems. In: 2017 IEEE Congress on Evolutionary Computation, CEC 2017 - Proceedings. Institute of Electrical and Electronics Engineers Inc., pp 372–379
21. Meng Z, Pan JS, Kong L (2018) Parameters with adaptive learning mechanism (PALM) for the enhancement of differential evolution. Knowl Based Syst 141:92–112. https://doi.org/10.1016/j.knosys.2017.11.015
22. Jadon SS, Tiwari R, Sharma H, Bansal JC (2017) hybrid artificial bee Colony algorithm with differential evolution. Appl Soft Comput J 58:11–24. https://doi.org/10.1016/j.asoc.2017.04.018
23. Elaziz MA, Xiong S, Jayasena KPN, Li L (2019) Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution. Knowl Based Syst 169:39–52. https://doi.org/10.1016/j.knosys.2019.01.023
24. Yildizdan G, Baykan ÖK (2020) A novel modified bat algorithm hybridizing by differential evolution algorithm. Expert Syst Appl. https://doi.org/10.1016/j.eswa.2019.112949
25. Zhang J, Sanderson AC (2009) JADE: adaptive differential evolution with optional external archive. IEEE Trans Evol Comput 13:945–958. https://doi.org/10.1109/TEVC.2009.2014613
26. Mohamed AW, Hadi AA, Fattouh AM, Jambi KM (2017) LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems. In: 2017 IEEE Congress on Evolutionary Computation, CEC 2017 - Proceedings. Institute of Electrical and Electronics Engineers Inc., pp 145–152
27. Ibrahim RA, Elaziz MA, Lu S (2018) Chaotic opposition-based grey-wolf optimization algorithm based on differential evolution and disruption operator for global optimization. Expert Syst Appl 108:1–27. https://doi.org/10.1016/j.eswa.2018.04.028
28. Segredo E, Lalla-Ruiz E, Hart E, Voß S (2018) On the performance of the hybridisation between migrating birds optimisation variants and differential evolution for large scale continuous problems. Expert Syst Appl 102:126–142. https://doi.org/10.1016/j.eswa.2018.02.024
29. De Melo VV, Iacca G (2014) A modified covariance matrix adaptation evolution strategy with adaptive penalty function and restart for constrained optimization. Expert Syst Appl 41:7077–7094. https://doi.org/10.1016/j.eswa.2014.06.032
30. Shuster JJ (2007) Design and analysis of experiments. Methods Mol Biol 404:235–259
31. Brest J, Maučec MS, Bošković B (2016) IL-SHADE: Improved L-SHADE algorithm for single objective real-parameter optimization. In: 2016 IEEE Congress on Evolutionary Computation, CEC 2016. Institute of Electrical and Electronics Engineers Inc., pp 1188–1195
32. Gol Alikhani M, Javadian N, Tavakkoli-Moghaddam R (2009) A novel hybrid approach combining electromagnetism-like method with Solis and Wets local search for continuous optimization problems J. Glob Optim 44:227–234. https://doi.org/10.1007/s10898-008-9320-z
33. Tseng LY, Chen C (2008) Multiple trajectory search for large scale global optimization. In: 2008 IEEE Congress on Evolutionary Computation, CEC 2008. pp 3052–3058
34. Liu J, Zhang S, Wu C et al (2016) A hybrid approach to constrained global optimization. Appl Soft Comput J 47:281–294. https://doi.org/10.1016/j.asoc.2016.05.021
35. Shi Z, Yang G, Xiao Y (2016) A limited memory BFGS algorithm for non-convex minimization with applications in matrix largest eigenvalue problem. Math Methods Oper Res 83:243–264. https://doi.org/10.1007/s00186-015-0527-8
36. Biglari F (2015) Dynamic scaling on the limited memory BFGS method. Eur J Oper Res 243:697–702. https://doi.org/10.1016/j.ejor.2014.12.050
37. Badem H, Basturk A, Caliskan A, Yuksel ME (2017) A new efficient training strategy for deep neural networks by hybridization of artificial bee colony and limited–memory BFGS optimization algorithms. Neurocomputing 266:506–526. https://doi.org/10.1016/j.neucom.2017.05.061
38. Burton RM (1985) Pointwise properties of convergence in probability. Stat Probab Lett 3:315–316. https://doi.org/10.1016/0167-7152(85)90063-X
39. Saunders IW, Ross SM (1985) Stochastic processes. J Am Stat Assoc 80:250. https://doi.org/10.2307/2288101
40. Wu, Guohua; Mallipeddi, Rammohan; Suganthan P Problem definitions and evaluation criteria for the CEC 2017 competition and special session on constrained single objective real-parameter optimization
41. Wu G, Mallipeddi R, Suganthan P Problem definitions and evaluation criteria for the CEC 2017 competition and special session on constrained single objective real-parameter optimization
42. Auger A, Hansen N (2011) CMA-ES: evolution strategies and covariance matrix adaptation. In: Genetic and evolutionary computation conference, GECCO'11 - Companion Publication. pp 991–1010
43. Knight JN, Lunacek M (2007) Reducing the space-time complexity of the CMA-ES. In: Proceedings of GECCO 2007: Genetic and Evolutionary Computation Conference. pp 658–665
44. Hansen N, Müller SD, Koumoutsakos P (2003) Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evol Comput 11:1–18. https://doi.org/10.1162/106365603321828970
45. Cui L, Li G, Zhu Z et al (2018) Adaptive multiple-elites-guided composite differential evolution algorithm with a shift mechanism. Inf Sci (Ny) 422:122–143. https://doi.org/10.1016/j.ins.2017.09.002
46. Mohamed AW, Hadi AA, Jambi KM (2019) Novel mutation strategy for enhancing SHADE and LSHADE algorithms for global numerical optimization. Swarm Evol Comput. https://doi.org/10.1016/j.swevo.2018.10.006
47. Kumar A, Misra RK, Singh D (2017) Improving the local search capability of Effective Butterfly Optimizer using Covariance Matrix Adapted Retreat Phase. In: 2017 IEEE Congress on Evolutionary Computation, CEC 2017 - Proceedings. Institute of Electrical and Electronics Engineers Inc., pp 1835–1842
48. Awad NH, Ali MZ, Suganthan PN, Reynolds RG (2016) An ensemble sinusoidal parameter adaptation incorporated with L-SHADE for solving CEC2014 benchmark problems. In: 2016 IEEE Congress on Evolutionary Computation, CEC 2016. Institute of Electrical and Electronics Engineers Inc., pp 2958–2965
49. Zhao F, Zhang L, Zhang Y et al (2020) An improved water wave optimisation algorithm enhanced by CMA-ES and opposition-based learning. Conn Sci 32:132–161. https://doi.org/10.1080/09540091.2019.1674247

50. Zhan ZH, Zhang J, Li Y, Shi YH (2011) Orthogonal learning particle swarm optimization. IEEE Trans Evol Comput 15:832–847. https://doi.org/10.1109/TEVC.2010.2052054
51. García S, Molina D, Lozano M, Herrera F (2009) A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization. J Heuristics 15:617–644. https://doi.org/10.1007/s10732-008-9080-4
52. Sadollah A, Bahreininejad A, Eskandar H, Hamdi M (2013) Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems. Appl Soft Comput J 13:2592–2612. https://doi.org/10.1016/j.asoc.2012.11.026
53. Shao Z, Pi D, Shao W (2019) A novel multi-objective discrete water wave optimization for solving multi-objective blocking flow-shop scheduling problem. Knowl Based Syst 165:110–131. https://doi.org/10.1016/j.knosys.2018.11.021
54. Bean JC (1994) Genetic algorithms and random keys for sequencing and optimization. ORSA J Comput 6:154–160. https://doi.org/10.1287/ijoc.6.2.154
55. Taillard E (1993) Benchmarks for basic scheduling problems. Eur J Oper Res 64:278–285. https://doi.org/10.1016/0377-2217(93)90182-M