Complex & Intelligent Systems (2023) 9:4439–4456
https://doi.org/10.1007/s40747-022-00957-6

ORIGINAL ARTICLE

Cooperative coevolutionary differential evolution with linkage measurement minimization for large-scale optimization problems in noisy environments

Rui Zhong[1] · Enzhi Zhang[1] · Masaharu Munetomo[2]

Received: 25 August 2022 / Accepted: 26 December 2022 / Published online: 17 January 2023

Abstract
Many optimization problems suffer from noise, and the noise combined with the large-scale attributes makes the problem complexity explode. Cooperative coevolution (CC) based on divide and conquer decomposes the problems and solves the sub-problems alternately, which is a popular framework for solving large-scale optimization problems (LSOPs). Many studies show that the CC framework is sensitive to decomposition, and the high-accuracy decomposition methods such as differential grouping (DG), DG2, and recursive DG (RDG) are extremely sensitive to sampling accuracy, which will fail to detect the interactions in noisy environments. Therefore, solving LSOPs in noisy environments based on the CC framework faces unprecedented challenges. In this paper, we propose a novel decomposition method named linkage measurement minimization (LMM). We regard the decomposition problem as a combinatorial optimization problem and design the linkage measurement function (LMF) based on Linkage Identification by non-linearity check for real-coded GA (LINC-R). A detailed theoretical analysis explains why our proposal can determine the interactions in noisy environments. In the optimization, we introduce an advanced optimizer named modified differential evolution with distance-based selection (MDE-DS), and the various mutation strategy and distance-based selection endow MDE-DS with strong anti-noise ability. Numerical experiments show that our proposal is competitive with the state-of-the-art decomposition methods in noisy environments, and the introduction of MDE-DS can accelerate the optimization in noisy environments significantly.

Keywords Cooperative coevolution (CC) · Linkage measurement minimization (LMM) · MDE-DS · Large-scale optimization problems (LSOPs) · Noisy environments

Introduction

Noise widely exists in the fitness evaluation of many problems [1–3], which can mislead the direction of optimization. In the past decade, many studies [4,5] on optimization problems in noisy environments have been published, and some strategies have been introduced to traditional evolutionary algorithms (EAs) to tackle the noise. Examples include explicit averaging [6], implicit averaging [7], Fourier transform [8], fitness estimation [9], and more. Most of the previous research focuses on relatively low-dimensional problems (up to 100-D), and a few studies on noisy problems with large-scale optimization problems (LSOPs) have been published. In fact, many noisy optimization problems are high-dimensional, such as parameters and structures optimization of deep neural networks [10] and subset selection [11].

The LSOPs in noisy environments contain challenges both on the scalability and robustness to noise, which make the difficulties of problem-solving explosive. The main reasons are the following aspects: (1) the complexity of the optimization problem increases, including the increase of dimensionality and the existence of the noise. (2) The search space of LSOPs increases exponentially with the increase of dimensionality, which is known as the curse of dimensionality [12]. (3) The computational cost of building a surrogate model is expensive, and the accuracy is also affected by noise and

✉ Rui Zhong
zhongrui19961230@gmail.com

1 Graduate School of Information Science and Technology, Hokkaido University, Kita Ward, Kita 14 Jonishi, Sapporo, Hokkaido 060-0814, Japan

2 Information Initiative Center, Hokkaido University, Kita Ward, Kita 11 Jonishi, Sapporo, Hokkaido 060-0811, Japan

the curse of dimensionality, which makes some algorithms limited [13,14].

Many algorithms have been proposed to overcome the challenge of the LSOPs, such as designing optimization operators to adapt the large-scale attributes [15], building surrogate models [16], and decomposing the problem [17], which is known as the cooperative coevolution (CC). In this paper, we apply the CC framework to solve LSOPs in noisy environments. This method is inspired by the divide and conquer, which has achieved great success in solving large-scale continuous [18], combinatorial [19], and constrained [20] problems.

How to decompose the LSOPs is the key to the successful implementation of the CC framework. Many studies [21,22] show that the CC framework is sensitive to problem decomposition strategies. Taking the linkage identification by non-linearity check for real-coded GA (LINC-R) [23] as a pioneer, many decomposition methods have been proposed. Differential grouping (DG) [24] first extends the identical mechanism of LINC-R to the 1000-D problem. Extend DG (XDG) [25] improves the shortage of DG in dealing with overlaps. DG2 [26] notices the high computational cost in DG and utilizes the transmissibility of separability to save the computational budget. Global DG (GDG) [27] regards the variable interactions matrix as the adjacency matrix of a graph and depth-first search or breadth-first search is applied to identify the interactions and formed the sub-problems. Recursive DG (RDG) [18] further reduces the computational cost by examining the interaction between a pair of sets of variables rather than a pair of variables, and forms the sub-problems recursively. Efficient RDG (ERDG) [28] uses the historical information on interaction identification to save the computational cost in redundant examinations and is more efficient than RDG. These decomposition methods are considered high-accuracy decomposition methods. These methods detect the interaction by determining the difference between fitness difference and threshold. However, they are extremely sensitive to the fidelity of observed objective value and will completely fail to detect the interactions in multiplicative noisy environments. We will explain the reason in the section "Challenges of DG-based decomposition methods in noisy environments".

In this paper, we propose a novel decomposition method named linkage measurement minimization (LMM), our proposal allows an automatic decomposition that treats the decomposition problem as a combinatorial optimization problem, and we design the linkage measure function (LMF) based on LINC-R as the objective function of combinatorial optimization. In addition, the advanced optimizer: MDE-DS is employed to optimize the sub-problems (MDE-DSCC-LMM). More specifically, the main contributions of this paper are as follows.

(1) Our proposal LMM provides a novel strategy to regard the decomposition problem as a combinatorial optimization problem, and the genetic algorithm is employed to actively search the interactions between decision variables. We mathematically explain the feasibility of LMF and its relationship with LINC-R. Theoretical analysis shows how our proposal detects interactions in noisy environments. In addition, we analyze the time complexity of LMM, and the fitness evaluation times (FEs) consumed in decomposition are controllable. And our proposal can be extended to decompose the higher dimensional, multi-objective, real-world problems with a limited computational budget.
(2) MDE-DS is applied as the optimizer for sub-problems and is well performed on various benchmark functions in noisy environments. The results in this paper further demonstrate that MDE-DS can accelerate cooperative coevolutionary optimization significantly.
(3) Numerical experiments demonstrate that LMM is competitive with some state-of-the-art decomposition methods for LSOPs in noisy environments, and the introduction of MDE-DS is efficient for sub-problems optimization. To the best of our knowledge, not much work has been reported on employing the CC framework to solve LSOPs in noisy environments.

The rest of the paper is organized as follows: the section "Preliminaries and related work" covers preliminaries, MDE-DS, a brief review of the state-of-the-art decomposition method, and reveals the challenges of DG-based decomposition methods in multiplicative noisy environments. The section "Our proposal:MDE-DSCC-LMM" provides a detailed introduction to our proposal, MDE-DSCC-LMM. The section "Numerical experiment and analysis" describes the experiments on CEC2013 LSGO Suite [29] in noisy environments and analyzes the experimental results. The section "Discussion" discusses the direction of our research in the future. Finally, the section "Conclusion" concludes the paper.

Preliminaries and related work

Preliminaries

Large-scale optimization problem

Without loss of generality, an LSOP can be defined as follows:

$$\min f(X)$$
$$\text{s.t.}: X \in \mathbb{R}^{,}$$

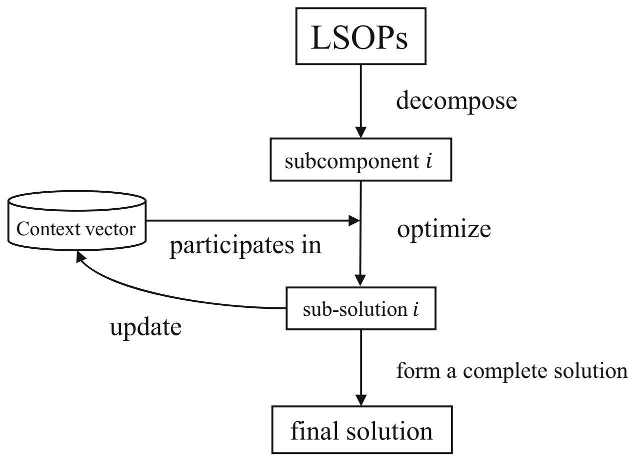

Fig. 1 The flowchart of CC

where $X = (x_1, x_2, \ldots, x_n)$ is an n-dimensional decision vector, and each x_i $(i \in [1, n])$ is a decision variable. $f(X)$ is the objective function needed to be minimized. In our work, the large-scale optimization problem is a special case of black-box optimization, where the number of decision variables n is large (e.g., $n \geq 1000$).

Variables' interaction

The concept of variable interaction is derived from biology. In biology, if a feature at the phenotype level is contributed by two or more genes, then we consider there are interactions between these genes, and the genome composed of these genes is called a linkage set [30]. In the definition of optimization problems. If $\min f(x_1, x_2, \ldots, x_n) = (\min_{c_1} f_1(..., ...), \ldots, \min_{c_m} f_m(..., ...))$, then $f(x)$ is a partially separable function, and decision variables in identical sub-problem consist of linkage set. There are two extreme cases, when there is no interaction between all variables, which means $\min\ f(x_1, x_2, \ldots, x_n) = \min \sum_{i=1}^{n} f(x_{i_1})$, then we call $f(x)$ is a fully separable function. On the contrary, we call $f(x)$ a completely nonseparable function if all variables have direct or indirect interactions with each other.

Cooperative coevolution

Inspired by divide and conquer, the CC framework was proposed to deal with LSOPs by decomposing the problem into multiple nonseparable sub-problems and optimizing them alternately. A standard CC consists of two stages: decomposition and optimization. Figure 1 shows the main steps of the CC framework.

CC framework first decomposes the LSOPs into k nonseparable sub-problems with a certain strategy. Due to the sub-solution $i (i \in [1, k])$ cannot form a complete solution for evaluation, all sub-problems maintain a public context vector [31] to construct a complete solution, and after optimization, the latest information updates the context vector. Some studies found that only one context vector may be too greedy for evaluation. Therefore, the adaptive multi-context CC framework [32] is proposed, which employs multiple context vectors to co-evolve subcomponents.

Noise in objective functions

Additive noise [33] and multiplicative noise [34] widely exist in the evaluation of optimization problems. Mathematically, the noisy objective function f^N(X) of a trial solution X is represented by

$$f^N(X) = f(X) + \eta \tag{1}$$
$$f^N(X) = f(X) \cdot (1 + \beta), \tag{2}$$

where $f(X)$ is the real objective function. Equation (1) shows the objective function in addictive noisy environments, η is the amplitude of the addictive noise. Equation (2) reveals the relationship between the real objective function and objective function in multiplicative noisy environments. β is a random noise (such as Gaussian noise).

Anti-noise strategies in EAs

Many optimization problems suffer from noise, and to perform the optimization under the existence of noise, various anti-noise strategies have been proposed in the literature. Following the classification reported in Ref. [35], two categories of noise handling methods for EAs can be mainly classified; each category can be divided into two sub-categories:

- Methods which require an increase in the computational cost
  (1) Explicit averaging methods
  (2) Implicit averaging methods.
- Methods which perform hypotheses about the noise
  (1) Averaging through approximated models
  (2) Modification of the selection schemes.

Explicit averaging methods consider that re-sampling and re-evaluation can reduce the impact of noise on the fitness landscape. Increasing the re-evaluation times is equivalent to reducing the variance of the estimated fitness. Thus, ideally, an infinite sample size would reduce to 0 uncertainties in the fitness estimations.

Implicit averaging states that a larger population allows the evaluations of neighbor solutions, and thus, the fitness landscape in a particular portion of decision space can be estimated. Paper [36] has shown that a large population size reduces the influence of noise on the optimization process, and paper [37] has proved that a GA with an infinite population size would be noise-insensitive.

Both explicit and implicit averaging methods consume more fitness evaluation times (FEs) to correct the objective value, which is improper or even unacceptable for LSOPs under the FEs' limitation. To obtain efficient noise filtering without excessive computational cost, various techniques have been proposed in the literature, such as the introduction of the approximated model [38], probability-based selection schemes [39], self-adaptative parameter adjustment [40], and so on.

Modified DE with distance-based selection

Differential evolution algorithm (DE) [41] was first proposed in 1995 and has been wildly applied in data mining [42], pattern recognition [43], artificial neural networks [44], and other fields due to its characteristics, such as easy implementation, fast convergence speed, and strong robustness. MDE-DS [45] is designed for continuous optimization problems in presence of noise with the modification in mutation, crossover, and selection, and the detailed description of MDE-DS is as follows.

Parameter control

The constants F and Cr are unnecessary, as F is randomly sampled from 0.5 to 2 for each mutation operation and Cr is randomly switched between 0.3 and 1 for each target vector. Switching F between two extreme corners of the feasible range is conducive to attaining a balance between exploration and exploitation of the search. And there is a new parameter b (the blending rate) in blending crossover, whose value is also randomly chosen from among three candidates: a low value of 0.1, a medium value of 0.5, and a high value of 0.9. The utility of such a switching scheme has been discussed in paper [46] for solving LSOPs.

Mutation

MDE-DS includes two different mutation strategies and switches them randomly with 50% probability.

In the population centrality-based mutation, the elite subpopulation (top 50%) is selected and $\vec{\tilde{X}}_{\text{best},G}$ is calculated by the arithmetic mean (centroid) of the subpopulation individuals. Eq. (3) is adopted to mutate the ith individual

$$\vec{V}_{i,G} = \vec{X}_{r1,G} + F\left(\vec{\tilde{X}}_{\text{best},G} - \vec{X}_{r2,G}\right), \tag{3}$$

where $\vec{X}_{r1,G}$ and $\vec{X}_{r2,G}$ are two different individuals corresponding to randomly chosen indices $r1$ and $r2$. $\vec{V}_{i,G}$ is the newly generated mutant vector corresponding to the current target vector for present generation G.

In the DMP-based mutation scheme, the best individual $\vec{X}_{\text{best},G}$ in each generation is selected and the dimension-wise average is implemented for both $\vec{X}_{\text{best},G}$ and the current target individual $\vec{X}_{i,G}$. The mutation is generated in the following way:

$$\vec{V}_{i,G} = \vec{X}_{i,G} + \Delta_m \cdot \left(\frac{\vec{M}_{i,G}}{\|\vec{M}_{i,G}\|}\right) \tag{4}$$

where $\Delta_m = (X_{\text{best}_{\text{dim}},G} - X_{i_{\text{dim}},G})$, with $X_{\text{best}_{\text{dim}},G} = \frac{1}{D}\sum_{k=1}^{D} x_{\text{best}_k,G}$ and $X_{i_{\text{dim}},G} = \frac{1}{D}\sum_{k=1}^{D} x_{i_k,G}$. $\frac{\vec{M}_{i,G}}{\|\vec{M}_{i,G}\|}$ is a unit vector with random direction.

The significance of the population centrality-based mutation scheme is that it balances greediness while still maintaining a certain extent of diversity. For example, it is less greedy than the DE/best/1 scheme, and hence, the probability of the optimization trapped in local optima is less. On the other hand, the DMP-based mutation scheme prefers exploration [47], and thus, in absence of any feedback about the nature of the function, an unbiased combination of these two methods is applied.

Crossover

Crossover plays an important role in generating promising offspring from two or more existing individuals within the function landscape. Blending crossover is employed in MDE-DS and described in Eq. (5)

$$u_{j,i,G} = \begin{cases} b \cdot x_{j,i,G} + (1-b) \cdot v_{j,i,G} \\ x_{j,i,G} \end{cases}, \tag{5}$$

where $u_{j,i,G}$ and $v_{j,i,G}$ are the jth dimensions of the trial and donor vectors, respectively, corresponding to the current index i in generation G and $x_{j,i,G}$ is the jth dimension of the current population individual $\vec{X}_{i,G}$. Blending recombination has one parameter b, which is randomly selected from 0.1, 0.5, and 0.9. The concrete analysis can be referred to in Ref. [45].

Selection

The canonical DE selects the offspring based on a simple greedy strategy. However, if the fitness landscape gets corrupted with noise, the greedy selection suffers a lot, because in this case, the original fitness of parent and offspring is unknown and it can be well nigh impossible to infer when an offspring is superior or inferior to its parent. Thus, the design of selection is the key to anti-noise. To handle the

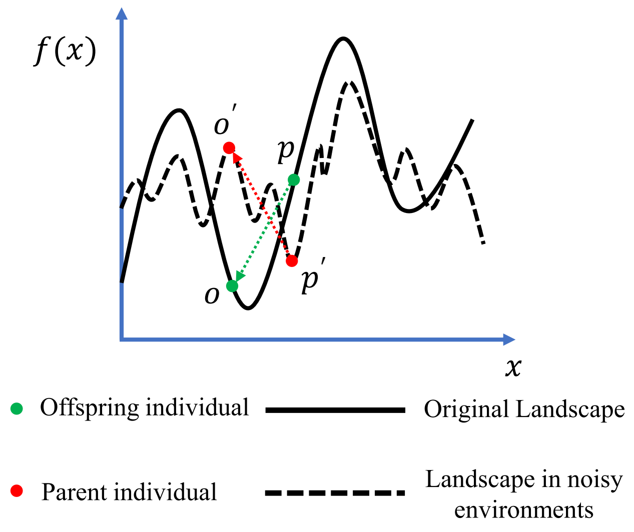

Fig. 2 A selection works on fitness landscape in noisy environments

presence of noise, a novel distance-based selection mechanism is introduced without any extra parameter. There are three cases of the proposed selection mechanism which are described subsequently

$$\vec{X}_{i,G+1} = \begin{cases} \vec{U}_{i,G}, \text{ if } \frac{f(\vec{U}_{i,G})}{f(\vec{X}_{i,G})} \leq 1 \\ \vec{U}_{i,G}, \text{ if } \frac{f(\vec{U}_{i,G})}{f(\vec{X}_{i,G})} > 1 \text{ and } p_s \leq e^{-\frac{\Delta f}{Dis}} \\ \vec{X}_{i,G}, \text{ else.} \end{cases} \tag{6}$$

In case 1, when $\frac{f(\vec{U}_{i,G})}{f(\vec{X}_{i,G})} \leq 1$, the offspring replaces the parent and survives to the next generation.

In case 2, although the parent performs better than the offspring, the offspring still can be preserved and survive into the next generation based on a stochastic principle. And the probability is calculated by $\mathrm{e}^{-\frac{\Delta f}{\mathrm{Dis}}}$, where $\Delta f = \left| f(\vec{U}_{i,G}) - f(\vec{X}_{i,G}) \right|$ represents the absolute fitness difference between $\vec{U}_{i,G}$ and $\vec{X}_{i,G}$, $\mathrm{Dis} = \sum_{k=1}^{D} \left| u_{i,k} - x_{i,k} \right|$ is the Manhattan distance between those two vectors. Manhattan distance is applied because of its simplicity and computational efficiency, and p_s is a random number generated from 0 to 1.

In case 3, If the parent significantly outperforms than offspring, then the offspring is removed and the parent persists to the next generation.

This selection process is further illustrated in Fig. 2.

Figure 2 shows a fitness landscape scenario both in noiseless environments and noisy environments, p and p' represent the parent individual in the original fitness landscape and landscape in noisy environments, o and o' represent the offspring individual in original fitness landscape and landscape in noisy environments, respectively. The fitness information we can observe is only in noisy environments, so in minimization problems, o' will be rejected to replace the p' in the next generation. The objective value of p is better than o in the real fitness landscape, and if we re-evaluate the o' and p', the domination may be changed. The mechanism of selection in MDE-DS allows the algorithm to give us some probabilistic flexibility to select worse solutions as in noise-affected landscapes.

In summary, the pseudocode of MDE-DS is shown in Algorithm 1

Algorithm 1 MDE-DS

```
Require: Population size : $s$; Maximum iteration : $T$; Dimension : $D$
Ensure: Optimized population : $X$; Fitness of population $X$ : $F$
1: $t \leftarrow 0$
2: $X \leftarrow$ initialPop($s$, $D$)
3: $F \leftarrow$ evaluate($X$)
4: while $t < T$ and not stop criterion do
5:   for $i = 0$ to $s$ do
6:     ▶ (Mutation)
7:     $r \leftarrow$ rand(0, 1)
8:     if $r \leq 0.5$ then
9:       update $\vec{V}_{i,G}$ by Eq. (3)
10:    else
11:      update $\vec{V}_{i,G}$ by Eq. (4)
12:    end if
13:    ▶ (Crossover)
14:    $Cr \leftarrow$ rand(0.3, 1)
15:    $b \leftarrow$ randChoice([0.1, 0.5, 0.9])
16:    for $j = 0$ to $D$ do
17:      update $u_{j,i,G} =$ by Eq. (5)
18:    end for
19:    ▶ (Selection)
20:    $F_i \leftarrow$ evaluate($U_i$)
21:    $\Delta f_i = \left| f(\vec{U}_i) - f(\vec{X}_i) \right|$
22:    $Dis = \sum_{k=1}^{D} \left| u_{i,k} - x_{i,k} \right|$
23:    $p_s \leftarrow$ rand(0, 1)
24:    choose $\vec{X}_{i,G+1}$ by Eq. (6)
25:    $i \leftarrow i + 1$
26:  end for
27:  $t \leftarrow t + 1$
28: end while
29: return $X$, $F$
```

A brief review of the state-of-the-art decomposition method

Based on the divide and conquer, the CC framework decomposes the LSOPs into multiple nonseparable sub-problems and optimizes them alternately, which is the mainstream framework for solving LSOPs. In this section, we will briefly review the state-of-the-art decomposition method.

Taking the LINC-R [30] as a pioneer, perturbation-based decomposition methods become one of the most popular strategies to collaborate with the CC framework. Equation (7) defines the perturbation in the ith dimension and the jth dimension

$$\begin{aligned} s &= (x_1, x_2, \ldots, x_n) \\ s_i &= (x_1, \ldots, x_i + \delta, \ldots, x_n) \\ s_j &= (x_1, \ldots, x_j + \delta, \ldots, x_n) \\ s_{ij} &= (x_1, \ldots, x_i + \delta, \ldots, x_j + \delta, \ldots, x_n). \end{aligned} \tag{7}$$

LINC-R identifies the interaction between variables based on the fitness difference of perturbation with pre-defined hyperparameter ε. More specifically

$$\begin{aligned} &\exists s \in \text{Pop}: \\ &\quad \text{if } |(f(s_{ij}) - f(s_i)) - (f(s_j) - f(s))| > \varepsilon, \\ &\quad \text{then } x_i \text{ and } x_j \text{ are nonseparable.} \end{aligned} \tag{8}$$

ε is the allowable error. DG extends Eq. (8) first to LSOPs up to 1000-D. Due to the FEs' limitation in LSOPs, the fitness difference from the lower bound of search space to the upper bound can be accepted. The reuse of fitness and negligence of indirect interactions decreases the needed FEs to $O(\frac{n^2}{m})$, and m is the number of sub-problems. In paper [24], a sensitivity test for threshold ϵ is also implemented, the experimental results show that the DG is sensitive to the threshold ϵ, and $\epsilon = 10^{-3}$ is a recommended value.

Subsequently, the extended DG (XDG) [25] noticed that DG cannot identify the overlapping; thus, it divides all direct and indirect interacting variables into a sub-problem, and then, the overlappings between sub-problems are checked to identify conditional interactions. The needed FEs of XDG are approximately n^2. The complexity of the XDG results in an unsuitable allocation of computational cost between decomposition and optimization and limits the development of XDG to deal with higher dimensional problems.

The high computational cost of decomposition is a critical problem. DG2 [26], a faster and more accurate DG-based decomposition method, was proposed to address this issue. DG2 utilizes the transmissibility of separability to save the FEs. For example, if x_1 interacts with x_2 and x_3, the identification between x_2 and x_3 is unnecessary, as they belong to the same sub-problem, and the computational cost of DG2 is reduced to $\frac{n^2+n+2}{2}$.

One of the most popular DG-based methods is Recursive DG (RDG) [48]. The RDG examines the interactions between a pair of sub-problems rather than a pair of single variables. For $f: \mathbb{R}^D \to \mathbb{R}$ is an objective function, $X_1 \subset X$ and $X_2 \subset X$ are two mutually exclusive subsets of variables: $X_1 \cap X_2 = \emptyset$. If there are two unit vectors $\mathbf{u}_1 \in U_{X_1}$ and $\mathbf{u}_2 \in U_{X_2}$, two real numbers $l_1, l_2 > 0$ and a solution $\mathbf{x}^*$ to satisfy Eq. (9)

$$f(\mathbf{x}^* + l_1 u_1 + l_2 u_2) - f(\mathbf{x}^* + l_2 u_2) \neq f(\mathbf{x}^* + l_1 u_1) - f(\mathbf{x}^*), \tag{9}$$

then there are some interactions between variables in X_1 and X_2; otherwise, X_1 and X_2 are considered as separable sets. If X_1 and X_2 interact with each other, and RDG divides X_2 into two equal-sized and mutually exclusive subsets, then interactions between X_1 and the two subsets are detected. Repeat the above process until RDG finds the variables which interact with X_1. The computational complexity of RDG is $O(n \log_2 n)$, which is better than DG, XDG, and DG2, and more friendly to higher dimensional problems.

The hyperparameter ε also plays an important role in interaction identification, and different problems have various fitness landscape characteristics, and the identical threshold may not be suitable for all problems. Inspired by DG2, RDG2 [49] introduces an upper bound of the round-off errors incurred by the calculation of the non-linearity term and applies it as the threshold value. The experimental results in Ref. [49] showed that RDG2 improves the accuracy of RDG in identifying the interactions between variables.

Challenges of DG-based decomposition methods in noisy environments

Additive noise and multiplicative noise are two representative noises. Additive noise is often irrelevant to the fitness landscape, so we can carefully adjust the parameters to overcome the additive noise in the decomposition, although it is not easy [50]. However, multiplicative noise is related to the fitness landscape, so fitness can amplify the noise. Dealing with multiplicative noise is more difficult than additive noise in the decomposition stage. Taking the LINC-R as an example

$$\begin{aligned} &\exists s \in \text{Pop}: \\ &\quad \Delta_1^N = f^N(s_i) - f^N(s) = f(s_i)(1+\beta_1) - f(s)(1+\beta_2) \\ &\quad \Delta_2^N = f^N(s_{ij}) - f^N(s_j) = f(s_{ij})(1+\beta_3) - f(s_j)(1+\beta_4) \\ &\quad \text{if } |\Delta_2^N - \Delta_1^N| > \varepsilon, \\ &\quad \text{then } x_i \text{ and } x_j \text{ are nonseparable} \end{aligned} \tag{10}$$

β_i is Gaussian noise. $|\Delta_2^N - \Delta_1^N| = |\Delta_2 - \Delta_1 + f(s_{ij})\beta_3 - f(s_j)\beta_4 - f(s_i)\beta_1 + f(s)\beta_2|$. When the noise $\beta \sim N(0, \sigma^2)$, we define the noise term $\phi_{ij} = f(s_{ij})\beta_3 - f(s_j)\beta_4 - f(s_i)\beta_1 + f(s)\beta_2$ which follows the distribution: $\phi_{ij} \sim N(0, (f^2(s_{ij}) + f^2(s_j) + f^2(s_j) + f^2(s))\sigma^2)$. In noisy environments with multiplicative noise, LINC-R cannot identify that the fitness difference is caused by interaction or noise and the probability of $\phi_{ij} = 0$ being satisfied is almost 0 [50]. In practice, the decomposition methods developed on the LINC-R, such as DG, DG2, RDG, etc. will fail in environments with multiplicative noise. We will provide experimental results of decomposition in the section "Performance of LMM". Therefore, grouping methods that detect interactions by perturbation face severe challenges.

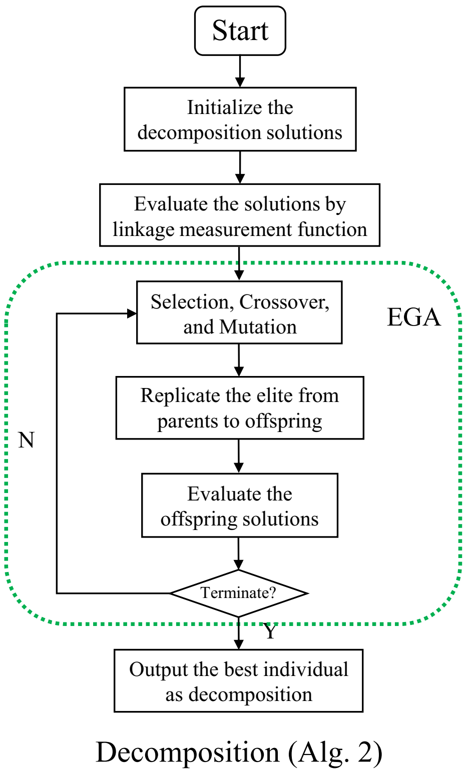
Start
Initialize the decomposition solutions
Evaluate the solutions by linkage measurement function
Selection, Crossover, and Mutation
EGA
Replicate the elite from parents to offspring
N
Evaluate the offspring solutions
Terminate?
Y
Output the best individual as decomposition
Decomposition (Alg. 2)

Fig. 3 The flowchart of decomposition (LMM)

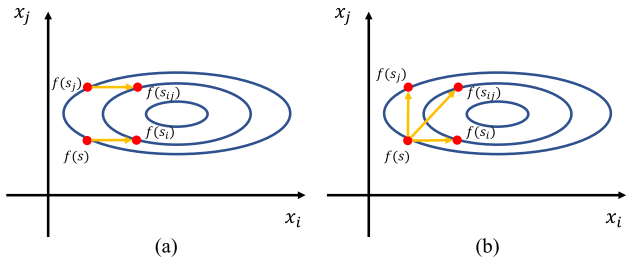

Fig. 4 **a** LINC-R works on the separable variables. **b** Variant LINC-R works on the separable variables

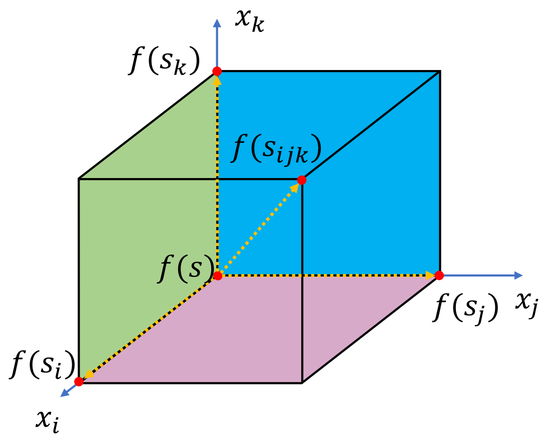

Fig. 5 The variant LINC-R works on 3-D space [53]

Our proposal: MDE-DSCC-LMM

In this section, we will introduce the details of our proposal. Our proposal consists of two stages: decomposition and optimization. In the decomposition, we divide the decision variables into sub-problems with our proposal: LMM, and in the optimization, MDE-DS is employed as a basic optimizer to optimize the sub-problems. Next, the concrete procedures of decomposition and optimization will be explained.

Decomposition: LMM

First, we provide the flowchart of our proposal in decomposition: LMM. The flowchart is shown in Fig. 3.

The basic idea is that we regard the decomposition problem as a combinatorial optimization problem and design the LMF based on LINC-R to lead the direction of searching for a better decomposition solution. The specific derivation of LMF is as follows.

The original LINC-R is defined as Eq. (11)

$\exists s \in$ Pop:

$$\text{if } |(f(s_{ij}) - f(s_j)) - (f(s_i) - f(s))| > \varepsilon, \quad (11)$$
$$\text{then } x_i \text{ and } x_j \text{ are nonseparable}$$

where the size of Pop is m, FEs consumed in a pair of variables based on Pop is $4m$, and the interaction between every pair of variables is identified in LINC-R. Thus, in the $n-$D problem, the necessary FEs is $2mn(n+1)$, which is unaffordable for LSOPs. Many studies [18,24,51] only detect the interactions by calculating the fitness difference from the lower bound of search space to the upper bound to save the FEs in decomposition, and we adopt the same strategy in our proposal, although it is not so robust and may fail to detect the interactions in trap functions [52].

We also notice that the original LINC-R can be transformed into the vector addition form. Equation (12) shows this variant LINC-R

$$\text{if } |(f(s_{ij}) - f(s)) - ((f(s_i) - f(s)) + (f(s_j) - f(s)))| < \varepsilon \text{ then } x_i \text{ and } x_j \text{ are separable.} \quad (12)$$

Figure 4 shows how LINC-R and the variant LINC-R work on the separable variables x_i and x_j. Although the form is different, the mechanisms of LINC-R and variant LINC-R are identical.

Based on this interesting finding, we derive LINC-R to 3-D and higher dimensions. In 3-D space, the schematic diagram is shown in Fig. 5.

Here, we define the fitness difference in 3-D

$$\begin{aligned}\Delta_i &= f(s_i) - f(s)\\ \Delta_j &= f(s_j) - f(s)\\ \Delta_k &= f(s_k) - f(s)\\ \Delta_{ijk} &= f(s_{ijk}) - f(s).\end{aligned} \tag{13}$$

When the variant LINC-R is applied simultaneously to determine the interactions between x_i, x_j, and x_k

$$\begin{aligned}&\text{if } |\Delta_{ijk} - (\Delta_i + \Delta_j + \Delta_k)| < \varepsilon,\\ &\text{then } x_i, x_j, \text{and } x_k \text{ are separable.}\end{aligned} \tag{14}$$

Therefore, we can reasonably infer that when the dimension reaches n

$$\begin{aligned}&\text{if } |\Delta_{1,2,\ldots,n} - (\Delta_1 + \Delta_2 + \cdots + \Delta_n)| < \varepsilon\\ &\text{then } x_1, x_2, \ldots, .x_n \text{ are separable.}\end{aligned} \tag{15}$$

However, when Eq. (15) is not satisfied, we only know that interactions exist in some variables, but we cannot know in which pairs of variables. Taking 3-D space as an example

$$\begin{aligned}&\text{if } |\Delta_{ijk} - (\Delta_i + \Delta_j + \Delta_k)| > \varepsilon \text{ and } |\Delta_{ijk} - (\Delta_{ij}\\ &+ \Delta_k)| < \varepsilon \text{ then } x_i, x_j \text{ are nonseparabale and } x_k\\ &\text{is separable from } x_i, x_j.\end{aligned} \tag{16}$$

Therefore, in the n-dimensional space, although it is difficult to detect the interactions between multiple variables through high-dimensional LINC-R directly, we can actively search for the interactions between variables through heuristic algorithms. According to the above description, in the n-dimensional problem, the linkage measurement function (LMF) is defined in Eq. (17)

$$\mathrm{LMF}(s) = \left(\Delta_{1,2,\ldots,n} - \sum_{i,j,\ldots}^{m} \Delta_{i,j,\ldots}\right)^2; \tag{17}$$

m is the number of sub-problems. LMF in noisy environments is defined in Eq. (18)

$$\mathrm{LMF}^{\mathrm{N}}(s) = \left(\Delta^N_{1,2,\ldots,n} - \sum_{i,j,\ldots}^{m} \Delta^N_{i,j,\ldots}\right)^2, \tag{18}$$

where $\Delta^N_{1,2,\ldots,n} = f^N(s_{1,2,\ldots,n}) - f^N(s) = f(s_{1,2,\ldots,n})(1 + \beta_i) - f(s)(1 + \beta_j)$. To optimize the $\mathrm{LMF}^{\mathrm{N}}(s)$, EGA is employed as the basic optimizer. Figure 6 demonstrates that how to decode from genotype to decomposition.

The length of a chromosome is LD, L is the genome length, and D is the dimension of the problem.

We decode the binary chromosome to decimal phenotype level and divide the decision variables into corresponding sub-problems, and the decision variables assigned to sub-problem 0 are regarded as separable variables. This procedure of optimization guided by LMF is named linkage measurement minimization (LMM), and the pseudocode of the decomposition is shown in Algorithm 2

Algorithm 2 The Pseudocode of decomposition

```
Require: Population size : $s$; Genome length : $L$; Maximum iteration : $T$ Dimension : $D$
Ensure: The best decomposition : $E$
1: ▶ (Decomposition solution initialization)
2: for $i = 0$ to $s_1$ do
3:     for $j = 0$ to $D$ do
4:         $n \leftarrow$ randint$(0, 2^{L-1} - 1)$
5:         $bn \leftarrow$ binary$(n)$
6:         $P^t_{i,j} \leftarrow bn$
7:     end for
8: end for
9: $F^t \leftarrow$ evaluate$(P^t)$
10: $E \leftarrow$ bestIndi$(P^t, F^t)$ # save the best solution
11: ▶ (Optimization by EGA)
12: while $t = 0$ and $t < T$ do
13:     $P^{t+1} \leftarrow$ selection$(P^t, F^t, s)$
14:     $P^{t+1} \leftarrow$ crossover$(P^{t+1})$
15:     $P^{t+1} \leftarrow$ mutation$(P^{t+1})$
16:     $F^{t+1} \leftarrow$ evaluate$(P^{t+1})$
17:     $P^{t+1} \leftarrow$ replace$(P^{t+1}, E)$ # elite is inherited to next generation
18:     $E \leftarrow$ bestIndi$(P^{t+1}, F_{t+1})$ # update the current best individual
19:     $t \leftarrow t + 1$
20: end while
21: return $E$
```

As the general process of GA, we first initialize the decomposition solutions randomly in Algorithm 2, from line 2 to 8. The object E saves the best decomposition solution. Then, we repeat the procedure of selection, crossover, mutation, evaluation, and inheritance until the iteration reaches the stop criterion from line 12 to 19. The elitist strategy [54] directly replicates the best individual to the next generation, which can prevent the elite individual from destroying the superior gene and chromosome structure during optimization.

Time complexity analysis

FEs consumed in interaction identification are analyzed in this section. As the structure of an individual in Fig. 6, the best and worst time complexity for evaluating an individual is $O(1)$ and $O(D)$, when all decision variables are identified as nonseparable and separable, respectively. D is the dimension of problems. Suppose that the population size is N, maximum iteration is M. Thus, the best and worst time complexity of our proposal LMM is $O(NM)$ and $O(DNM)$.

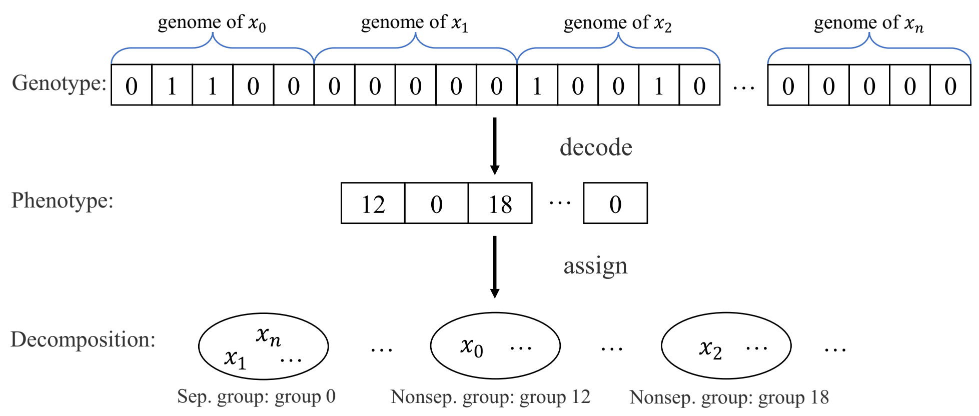

Fig. 6 A demonstration of decoding from genotype to decomposition

Theoretical support for LMM in noisy environments

It is evident that the optimization guided by LMF can identify the interactions in the noiseless environment, because the individuals containing correct linkage information have lower linkage measurement values and higher fitness, which prefer to survive in the selection of EGA. An example we mentioned before is Eq. (16). And an important explanation is why LMM can identify the interactions in noisy environments. Here, we provide theoretical support.

Corollary *Let x_i and x_j be separable decision variables, and x_m and x_n be nonseparable decision variables. $I(x_i, x_j) = (f(s_{ij}) - f(s_i)) - (f(s_j) - f(s))$ and $I^N(x_i, x_j) = (f^N(s_{ij}) - f^N(s_i)) - (f^N(s_j) - f^N(s))$ represent the intensity of interaction between x_i and x_j in noiseless environment and noisy environments, respectively. In noisy environments, if we prove that the probability $P(I^N(x_m, x_n) > I^N(x_i, x_j)) > 0$, which means it is possible that the intensity of the interaction between nonseparable variables can be stronger than separable variables in noisy environments, then the minimization of LMF can guide the direction to search for more interactions.*

Proposition *In noisy environments, the probability $P(I^N(x_m, x_n) > I^N(x_i, x_j)) > 0$, and individuals containing correct detected interactions have better fitness to survive.*

Proof In noisy environments, the noise $\beta \sim N(0, \sigma^2)$. The relationship between $I^N(\cdot)$ and $I(\cdot)$ is defined in Eq. (19)

$$I^N(x_i, x_j) = I(x_i, x_j) + (\beta_1 f(s_{ij}) - \beta_2 f(s_i)) - (\beta_3 f(s_j) - \beta_4 f(s)) = I(x_i, x_j) + \phi_{ij}, \tag{19}$$

where $\phi_{ij} = (\beta_1 f(s_{ij}) - \beta_2 f(s_i)) - (\beta_3 f(s_j) - \beta_4 f(s))$, and ϕ_{ij} follows the distribution:

$$\phi_{ij} \sim N(0, (f^2(s_{ij}) + f^2(s_j) + f^2(s_j) + f^2(s))\sigma^2), \tag{20}$$

and $I^N(x_i, x_j)$ follows the distribution:

$$I^N(x_i, x_j) \sim N(I(x_i, x_j), (f^2(s_{ij}) + f^2(s_j) + f^2(s_j) + f^2(s))\sigma^2). \tag{21}$$

Due to x_i and x_j are separable variables, x_m and x_n are nonseparable variables; similarly

$$I^N(x_i, x_j) \sim N(0, (f^2(s_{ij}) + f^2(s_j) + f^2(s_j) + f^2(s))\sigma^2) I^N(x_m, x_n) \sim N(I(x_m, x_n), (f^2(s_{mn}) + f^2(s_m) + f^2(s_n) + f^2(s))\sigma^2). \tag{22}$$

Here, we introduce a distribution $Y = I^N(x_m, x_n) - I^N(x_i, x_j)$, and the problem is transformed to prove $P(Y > 0) > 0$. Y follows the distribution:

$$Y \sim N(I(x_m, x_n), (f^2(s_{ij}) + f^2(s_j) + f^2(s_j) + f^2(s))\sigma^2 + (f^2(s_{mn}) + f^2(s_m) + f^2(s_n) + f^2(s))\sigma^2) \sim N(I(x_m, x_n), \sigma_{ijmn}). \tag{23}$$

The expectation of Y is $I(x_m, x_n)$, and there are two cases that need to be discussed:

Case 1: $I(x_m, x_n) > 0$: In this case, $P(Y > 0) > 0.5$.

Case 2: $I(x_m, x_n) < 0$: In this case, $0 < P(Y > 0) < 0.5$.

In summary, $P(I^N(x_m, x_n) > I^N(x_i, x_j)) > 0$ is true, and the optimization of LMF has the probability to detect more interactions in noisy environments, which can be employed as the objective function in our experiment. □

Optimization: MDE-DSCC

Figure 7 shows the procedure of optimization.

In the optimization, we first introduce the decomposition from Algorithm 2 to divide the decision variables into k subproblems, and an empty set of the context vector is initialized.

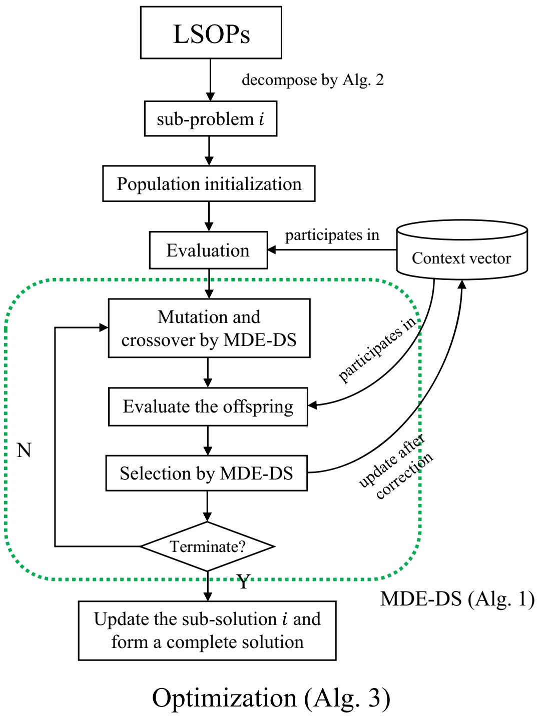

Fig. 7 The flowchart of optimization (MDE-DSCC)

For each sub-problem $i (i \in [1, k])$, we alternately optimize it with MDE-DS. The pseudocode of the whole optimization stage is shown in Algorithm 3.

Algorithm 3 Pseudocode of optimization

```
Require: Dimension : $D$; Computational budget : $C$; Population size : $M$ Maximum iteration : $T$
Ensure: The best solution : $E$
1: $SP \leftarrow$ LMM() # Alg. 2 decomposes the original problem
2: ▶ (Initialization)
3: $V \leftarrow [0] * D$ # empty context vector
4: for $i = 0$ to len($SP$) do
5:     $P_i^t \leftarrow$ initialPop($SP_i, M$)
6:     $O_i^t \leftarrow$ evaluate($SP_i, V$)
7:     $E \leftarrow$ bestIndi($P_i^t, O_i^t$)
8:     $V \leftarrow$ update($V, E$)
9: end for
10: ▶ (Optimization)
11: while $t = 0$ and $t < T$ do
12:     for $i = 0$ to len($SP$) do
13:         $P_i^{t+1}, O_i^{t+1} \leftarrow$ MDE-DS($P_i^t, O_i^t, V$) # Alg. 1
14:         $E \leftarrow$ bestIndi($P_i^{t+1}, O_i^{t+1}$)
15:         $V \leftarrow$ update($V, E$)
16:     end for
17:     $t \leftarrow t + 1$
18: end while
19: return $E$
```

In Algorithm 3, the initialization of optimization is executed from line 3 to 10. Here, we randomly generate the sub-populations for each sub-problem and update the context vector after evaluating the sub-populations. Then, sub-problems are optimized alternately from line 12 to 20 until all FEs consumed. The context vector is updated after every generation of optimization is finished.

Numerical experiment and analysis

In this section, a set of experiments are implemented to evaluate our proposal, MDE-DSCC-LMM. In the section “Experiment settings”, we introduce the experiment settings, including benchmark functions, comparing methods, and performance indicators. In the section “Performance of our proposal: MDE-DSCC-LMM”, we provide the experimental results of our proposal and comparing methods. Finally, we analyze our proposal both in the decomposition and optimization in the section “Analysis”.

Experiment settings

Benchmark functions

We design 15 test functions in noisy environments based on CEC2013 LSGO Suite, and Eq. (24) defines the benchmark functions in our experiments

$$f_i^N(x) = f_i(x) \cdot (1 + \beta), \quad i \in [1, 15] \tag{24}$$

$\beta \sim N(0, 0.01)$. Briefly, this benchmark suite consists of 15 test functions with 4 categories.

(1) $f_1^N(x)$ to $f_3^N(x)$: fully separable functions in noisy environments;
(2) $f_4^N(x)$ to $f_7^N(x)$: partially separable functions with 7 none-separable parts in noisy environments;
(3) $f_8^N(x)$ to $f_{11}^N(x)$: partially separable functions with 20 none-separable parts in noisy environments;
(4) $f_{12}^N(x)$ to $f_{15}^N(x)$: functions with overlapping sub-problems in noisy environments;
f_{13} and f_{14} consist of 905 decision variables, and the rest functions are 1000-D problems.

Comparing methods and parameters

In our experiment design, we compare the decomposition strategy of our proposal with various grouping methods, and the algorithms applied in the comparisons are listed in Tables 1 and 2 shows the parameters of our proposal in the decomposition stage. We also conduct the experiment between MDE-DSCC-LMM and DECC-LMM to show the effect of the introduction of MDE-DS. The maximum FEs including

Table 1 A summary of the algorithms under comparison

Algorithms	Decomposition methods
DECC-D	Delta grouping [55]
DECC-G	Random grouping [56]
DECC-DG	DG [51]
DECC-RDG	RDG [18]
DECC-DG2	DG2 [26]
DECC-LMM	LMM
MDE-DSCC-LMM	

decomposition and optimization are 3,000,000, and the population size of optimization for each sub-problem is set to 30.

Performance indicators

There are two stages of our proposal that need to be evaluated: LMM and MDE-DSCC.

To evaluate the LMM, three metrics are employed: FEs consumed in decomposition, decomposition accuracy (DA), and optimization results. We adopt the calculation method

Table 3 The detailed decomposition results of DG, DG2, and RDG on CEC2013 LSGO Suite in noisy environments

Func	Sep. vars	Nonsep. vars	Nonsep. groups	DG($\varepsilon = 10^{-3}$)/DG2/RDG($\alpha = 10^{-10}$)		
				Grouped sep. vars	Grouped nonsep. vars	Formed group's number
f_1	1000	0	0	0/0/0	1000/1000/1000	1/1/1
f_2	1000	0	0	0/0/0	1000/1000/1000	1/1/1
f_3	1000	0	0	0/0/0	1000/1000/1000	1/1/1
f_4	700	300	7	0/0/0	1000/1000/1000	1/1/1
f_5	700	300	7	0/0/0	1000/1000/1000	1/1/1
f_6	700	300	7	0/0/0	1000/1000/1000	1/1/1
f_7	700	300	7	0/0/0	1000/1000/1000	1/1/1
f_8	0	1000	20	0/0/0	1000/1000/1000	1/1/1
f_9	0	1000	20	0/0/0	1000/1000/1000	1/1/1
f_{10}	0	1000	20	0/0/0	1000/1000/1000	1/1/1
f_{11}	0	1000	20	0/0/0	1000/1000/1000	1/1/1
f_{12}	0	1000	1	0/0/0	1000/1000/1000	1/1/1
f_{13}	0	905	1	0/0/0	905/905/905	1/1/1
f_{14}	0	905	1	0/0/0	905/905/905	1/1/1
f_{15}	0	1000	1	0/0/0	1000/1000/1000	1/1/1

Table 4 The DA and consumed FEs of DG, RDG, DG2, and LMM on CEC2013 LSGO Suite in noisy environments

Func	DG		DG2		RDG		LMM	
	DA	FEs	DA	FEs	DA	FEs	DA	FEs
f_1	–	2.00e+03	–	5.01e+05	–	6.16e+03	–	6.14e+04
f_2	–	2.00e+03	–	5.01e+05	–	6.16e+03	–	6.21e+04
f_3	–	2.00e+03	–	5.01e+05	–	6.16e+03	–	6.17e+04
f_4	10.0%	2.00e+03	10.0%	5.01e+05	10.0%	6.16e+03	**16.1%**	5.82e+04
f_5	10.0%	2.00e+03	10.0%	5.01e+05	10.0%	6.16e+03	**15.8%**	5.82e+04
f_6	10.0%	2.00e+03	10.0%	5.01e+05	10.0%	6.16e+03	**16.4%**	5.82e+04
f_7	10.0%	2.00e+03	10.0%	5.01e+05	10.0%	6.16e+03	**17.1%**	5.82e+04
f_8	10.0%	2.00e+03	10.0%	5.01e+05	10.0%	6.16e+03	**14.1%**	5.12e+04
f_9	10.0%	2.00e+03	10.0%	5.01e+05	10.0%	6.16e+03	**13.4%**	4.56e+04
f_{10}	10.0%	2.00e+03	10.0%	5.01e+05	10.0%	6.16e+03	**13.8%**	4.67e+04
f_{11}	10.0%	2.00e+03	10.0%	5.01e+05	10.0%	6.16e+03	**14.2%**	4.12e+04
f_{12}	**100%**	2.00e+03	**100%**	5.01e+05	**100%**	6.16e+03	11.9%	4.80e+04
f_{13}	–	1.81e+03	–	4.09e+05	–	5.43+03	–	4.66e+04
f_{14}	–	1.81e+03	–	4.09e+05	–	5.43+03	–	4.76e+04
f_{15}	**100%**	2.00e+03	**100%**	5.01e+05	**100%**	6.16e+03	12.7%	4.80e+04

Table 2 The parameters of decomposition optimization

Parameter	Value
Optimization direction	Minimization
Optimizer	EGA
Population size	10
Maximum iteration	100
Genome length	5

of the DA in [49]. Essentially, DA is the ratio of the number of interacting variables that are correctly grouped to the total number of interacting variables. And to determine the existence of significance, we apply the Kruskal–Wallis test to the fitness at the end of the optimization in 25 trial runs between different decomposition methods. If significance exists, then we apply the p value acquired from the Mann–Whitney U test to do the Holm test. If LMM is significantly better than the second-best algorithm, we mark * (significance level 5%) or ** (significance level 10%) in the convergence curve.

To evaluate the MDE-DS, we apply the Mann–Whitney U test between MDE-DSCC-LMM and DECC-LMM. If MDE-DSCC-LMM is significantly better than DECC-LMM, we mark *(significance level 5%) or **(significance level 10%) at the end of optimization.

Performance of our proposal: MDE-DSCC-LMM

In this section, the performance of MDE-DSCC-LMM is studied, both on the decomposition and optimization. Experiments are conducted on the benchmark functions presented in the section “Benchmark functions”.

Performance of LMM

To verify the analysis in the section “Challenges of DG-based decomposition methods in noisy environments” that DG-based decomposition methods cannot detect the interactions in noisy environments, we apply DG, DG2, and RDG to decompose the benchmark functions. Table 3 shows the decomposition results.

The decomposition results of DG-based methods prove our analysis, all variables are divided into a sub-problem, and interactions failed to be detected completely. Next, we provide the DA and FEs for the decomposition of DG, RDG, DG2, and LMM in Table 4, because the decomposition results of LMM are different in every trial run, the DA and consumed FEs are calculated with the mean of 25 trial runs. The best DA is in bold.

Finally, the optimization results of DECC-D, DECC-G, DECC-DG, DECC-DG2, DECC-RDG, and DECC-LMM are provided in Table 5, and the best solution is in bold.

Performance of MDE-DSCC

The mean and standard deviation of the optimum between DECC-LMM and MDE-DSCC-LMM are shown in Table 6.

And the convergence curve of 25 independent runs of all compared methods is shown in Fig. 8.

Table 5 Optimization results of DECC-D, DECC-G, DECC-DG, DECC-DG2, DECC-RDG, and DECC-LMM

Func	DECC-D		DECC-G		DECC-DG		DECC-DG2		DECC-RDG		DECC-LMM	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
f_1	1.52e+08	2.81e+07	1.49e+08	2.9e+07	1.92e+09	2.81e+08	2.44e+09	2.56e+08	2.00e+09	2.62e+08	**6.10e+07**	**1.65e+07**
f_2	8.10e+04	3.30e+03	7.89e+04	3.19e+03	9.16e+04	3.35e+03	9.24e+04	2.43e+03	9.09e+04	3.43e+03	**5.65e+04**	**9.39e+03**
f_3	2.11e+01	8.01e−02	2.12e+01	5.07e−02	2.11e+01	1.07e−01	2.12e+01	8.75e−02	2.11e+01	1.11e−01	**2.11e+01**	**8.62e−02**
f_4	1.48e+12	6.30e+11	1.38e+12	6.36e+11	9.69e+11	2.41e+11	1.12e+12	2.34e+11	8.71e+11	2.31e+11	**6.75e+11**	**2.31e+11**
f_5	1.36e+07	1.47e+06	1.27e+07	1.60e+06	1.53e+07	1.15e+06	1.54e+07	1.30e+06	1.56e+07	1.73e+06	**1.06e+07**	**1.03e+06**
f_6	1.048e+06	2.98e+03	1.049e+06	3.33e+03	1.048e+06	3.16e+03	1.051e+06	3.70e+03	1.050e+06	3.94e+03	**1.044e+06**	**5.88e+03**
f_7	3.77e+09	1.65e+09	3.70e+09	1.06e+09	3.40e+09	7.28e+08	3.84e+09	1.21e+09	4.04e+09	6.31e+08	**2.88e+09**	**1.14e+09**
f_8	2.73e+16	1.20e+16	3.26e+16	2.13e+16	4.76e+16	1.79e+16	5.16e+16	1.84e+16	4.87e+16	2.07e+16	**1.99e+16**	**5.33e+15**
f_9	9.41e+08	1.86e+08	9.38e+08	1.81e+08	1.15e+09	1.16e+08	1.20e+09	7.30e+07	1.14e+09	9.42e+07	**8.22e+08**	**1.07e+08**
f_{10}	9.33e+07	3.98e+05	9.31e+07	4.12e+05	9.31e+07	3.79e+05	9.33e+07	4.43e+05	9.32e+07	5.21e+05	**9.21e+07**	**6.88e+05**
f_{11}	4.41e+11	1.63e+11	5.46e+11	2.40e+11	4.57e+11	1.63e+11	5.05e+11	1.26e+11	4.77e+11	1.23e+11	**2.54e+11**	**8.98e+10**
f_{12}	5.82e+12	2.68e+11	5.88e+12	2.05e+11	6.54e+12	2.46e+11	6.55e+12	2.01e+11	6.47e+12	2.06e+11	**3.55e+12**	**3.59e+11**
f_{13}	2.79e+10	9.27e+09	2.63e+10	5.61e+09	3.69e+10	6.90e+09	4.11e+10	7.81e+09	3.92e+10	8.94e+09	**1.81e+10**	**3.42e+09**
f_{14}	5.94e+11	1.69e+11	5.60e+11	1.47e+11	6.96e+11	1.51e+11	7.97e+11	1.45e+11	7.15e+11	1.17e+11	**3.68e+11**	**6.65e+10**
f_{15}	4.34e+07	7.92e+06	4.04e+07	5.55e+06	6.03e+07	1.42e+07	7.05e+07	1.29e+07	5.65e+07	8.69e+06	**2.20e+07**	**2.37e+06**

Table 6 Optimization results between DECC-LMM and MDE-DSCC-LMM

Func	DECC-LMM		MDE-DSCC-LMM	
	Mean	Std	Mean	Std
f_1	6.10e+07	1.65e+07	**4.95e+06**	**2.49e+06**
f_2	5.65e+04	9.39e+03	**1.48e+04**	**9.15e+02**
f_3	2.11e+01	8.62e−02	**2.09e+01**	**4.02e−02**
f_4	6.75e+11	2.31e+11	**2.75e+11**	**1.64e+11**
f_5	1.06e+07	1.03e+06	**9.79e+06**	**1.63e+06**
f_6	1.044e+06	5.88e+03	**1.040e+06**	**2.72e+03**
f_7	2.88e+09	1.14e+09	**1.08e+09**	**3.03e+08**
f_8	1.99e+16	5.33e+15	**5.39e+15**	**1.53e+15**
f_9	**8.22e+08**	**1.07e+08**	8.80e+08	9.42e+07
f_{10}	**9.21e+07**	**6.88e+05**	9.39e+07	4.77e+05
f_{11}	2.54e+11	8.98e+10	**1.61e+11**	**8.12e+10**
f_{12}	3.55e+12	3.59e+11	**5.11e+08**	**1.27e+08**
f_{13}	1.81e+10	3.42e+09	**1.17e+10**	**3.10e+09**
f_{14}	3.68e+11	6.65e+10	**2.59e+11**	**7.79e+10**
f_{15}	2.20e+07	2.37e+06	**9.58e+06**	**1.44e+06**

Analysis

In this section, we will analyze the performance of LMM and MDE-DS.

LMM in noisy environment

Theoretical analysis in the section "Theoretical support for LMM in noisy environments" shows that LMM has the potential to correctly detect the interactions between decision variables in noisy environments. Experimental results in the section "Performance of LMM" further support this analysis. The identification of interactions in noisy environments is a difficult task, and LMM identifies the decision variables with relatively strong intensity as nonseparable. Although the fitness difference will be affected by noise, the relative intensity of interactions between separable variables and nonseparable variables still has a possible gap, which is the main reason for successful implementation in noisy environments.

However, the optimization of LMF is not an easy task. From the DA in Table 4, the interactions which can be detected by LMM in noisy environments are limited. Although LMF can lead the direction of optimization to search for more correct interactions, a more powerful optimizer will allow LMM to find more interactions.

LMM vs DG-based decomposition methods

DG-based decomposition methods detect the interactions by determining the difference between the fitness difference and a certain parameter ϵ, and even in fully separable functions, the fitness difference will be amplified by noise and larger than ϵ easily, which is the main reason of detection failure in noisy environments, and all decision variables are divided into a sub-problem and optimized directly. Due to the curse of dimensionality, it is difficult for DE to find an acceptable solution with this division. Thus, although the DA of DG-based methods is higher than LMM in f_{12} and f_{15}, LMM still performed better than DG-based methods in the optimization of these functions, and DG-based methods are the most environmentally sensitive grouping method among the compared methods.

LMM vs D

The schematic diagram of Delta Grouping is shown in Fig. 9.

Delta grouping notices that the difference in coordinates from the initial random population to the optimized population is different in the separable variables and the non-separable variables. In Fig. 9, when the Δ_i and Δ_j has large difference, Delta grouping identify x_i and x_j are separable. This rough estimation is still affected by the noise, because the Delta grouping samples in the fitness landscape and the moving vector will still be influenced by the noise. Thus, Delta grouping is second sensitive in our comparing methods, and experimental results from Table 4 and Fig. 8 all show that our proposed LMM outperforms DECC-D.

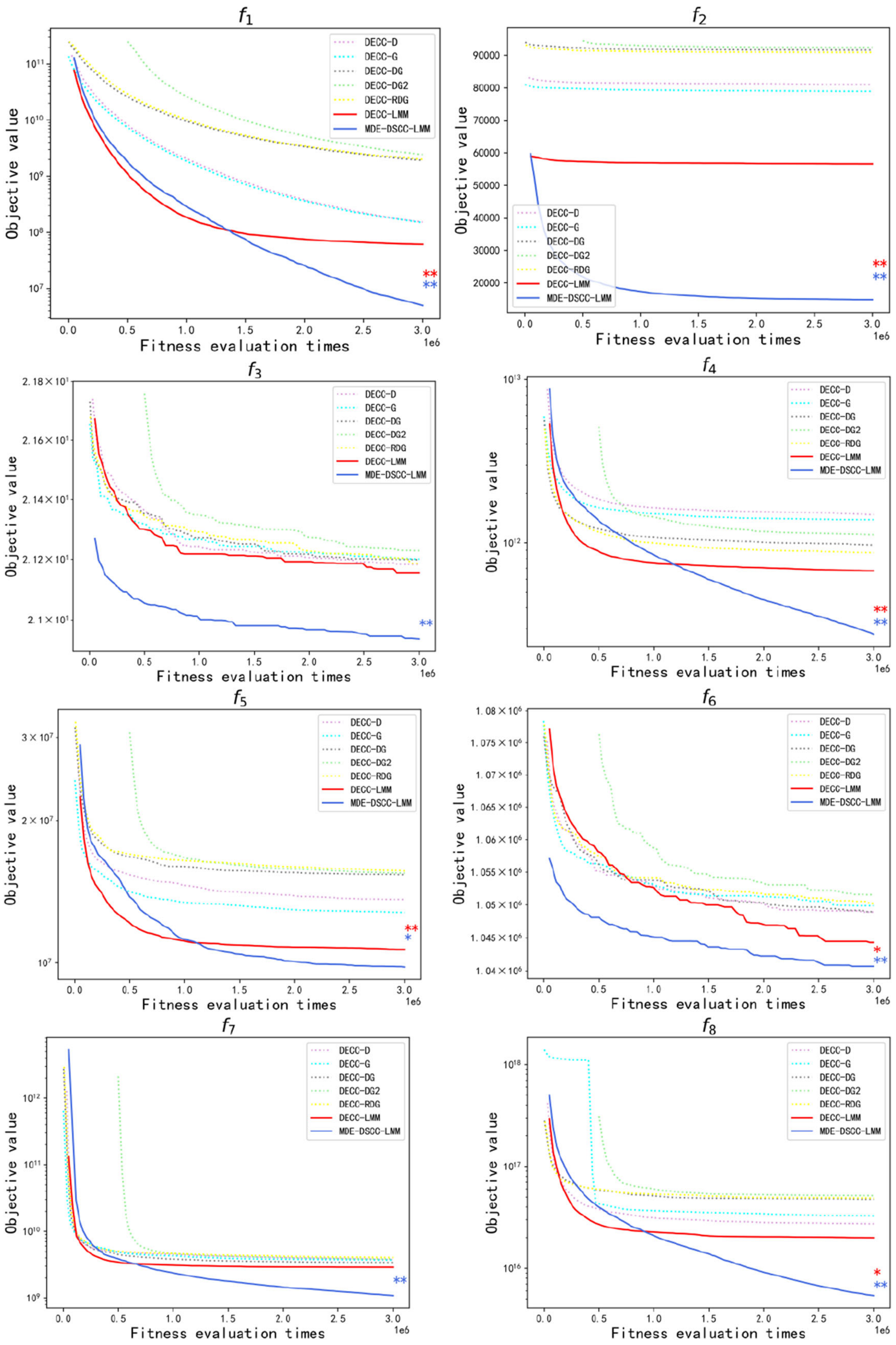

Fig. 8 The convergence curve of DECC-D, DECC-G, DECC-DG, DECC-DG2, DECC-RDG, DECC-LMM, and MDE-DSCC-LMM. The gap in the initial period is FEs consumed for decomposition

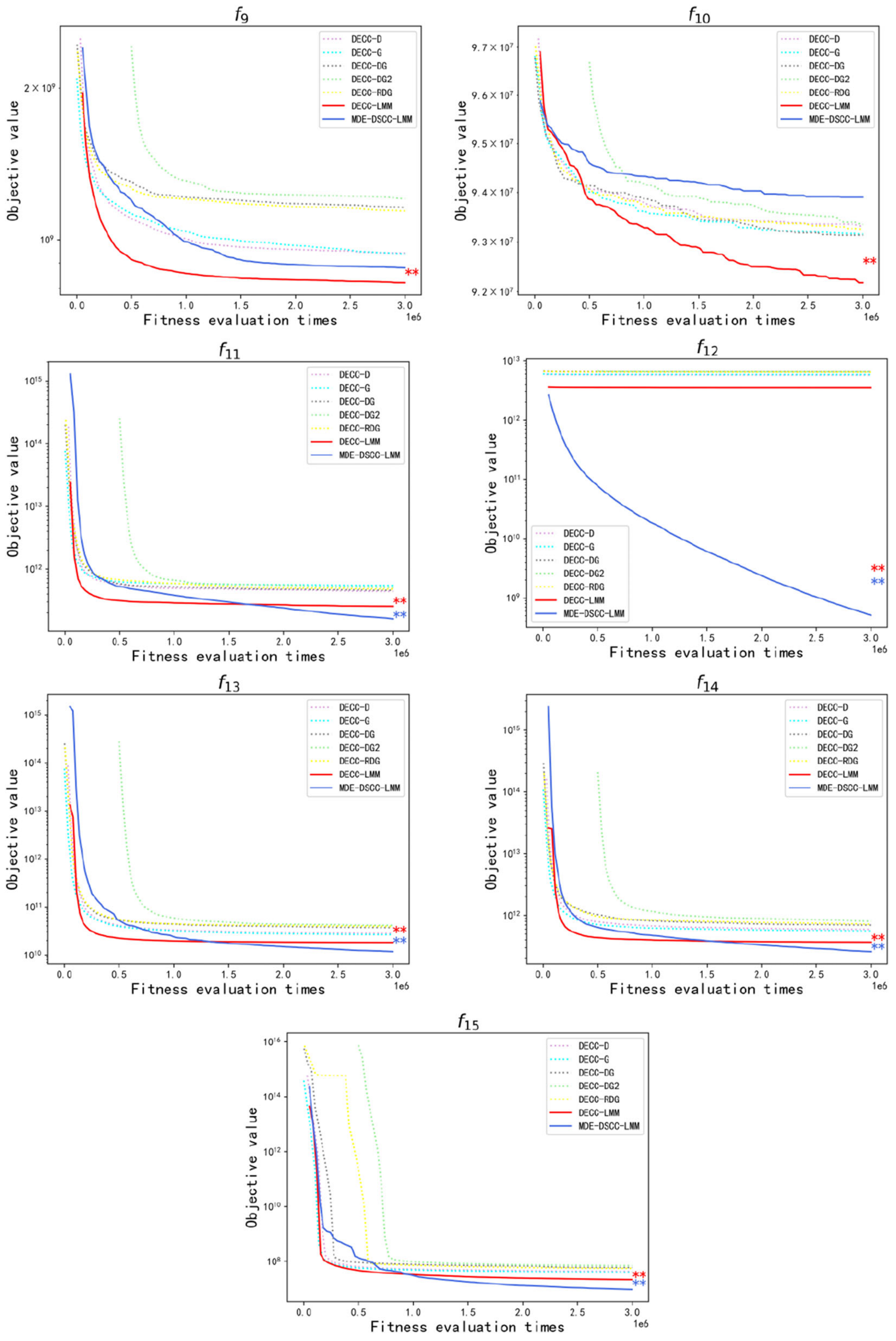

Fig. 8 continued

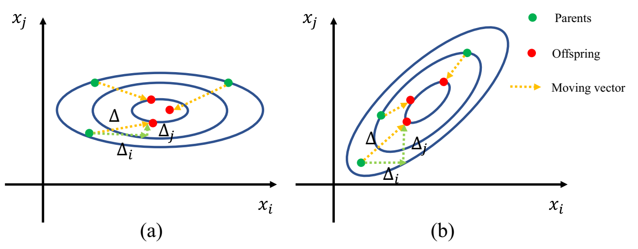

Fig. 9 **a** Delta grouping works on the separable function. **b** Delta grouping works on the nonseparable function

LMM vs random grouping

It is unnecessary to provide any information about the fitness landscape to Random grouping; therefore, Random grouping is the most environmentally insensitive decomposition method. Although paper [56] has proven that Random grouping is efficient and has a high probability to capture some interactions, it cannot detect sufficient interactions and form sub-problems properly, and LMM can detect more corrected interactions, which is the main reason that LMM outperformed than Random grouping.

The efficiency of MDE-DS

Figure 8 and Table 6 all prove that MDE-DS has a strong ability to search for better solutions compared with the canonical DE in most benchmark functions, although canonical DE performs better in f_9 and f_{10}. However, no optimization algorithm can solve all optimization problems perfectly. According to no free lunch theory [57] in optimization, the average performance of any pair of algorithms A and B is identical on all possible problems. Therefore, if an algorithm performs well on a certain class of problems, it must pay for that with performance degradation on the remaining problems, since this is the only way for all algorithms to have the same performance on average across all functions. Thus, although MDE-DS may perform worse than the canonical DE in noiseless functions, it is successful to introduce MDE-DS to solve problems in noisy environments.

Discussion

The above experimental results and analysis show that our proposal both the LMM and the introduction of MDE-DS have broad prospects to solve LSOPs in noisy environments. However, there are still many aspects for improvement. Here, we list some open topics for potential and future research.

How to improve the LMM

In this paper, we regard the decomposition problem as a combinatorial optimization problem and design the LMF to guide the direction of optimization by EGA. There are two parts of LMM that can be improved: (1). The design of LMF and (2). Optimizer for LMF. For LMF, we notice that it is multi-modal, especially for separable functions. For example, $f(x) = 2x_1 + x_2^2 - 0.5\sqrt{x_3}$, $((x_1, x_2, x_3))$, $((x_1, x_2), x_3))$, $((x_1, x_3), x_2))$, $((x_1), (x_2, x_3))$, $((x_1), (x_2), (x_3))$ are all global optima, and actually, $((x_1), (x_2), (x_3))$ is our ideal decomposition. Thus, how to design the LMF to avoid this issue is a problem that can be improved in future research. And for the optimizer, this paper employed EGA to optimize the LMF, and parts of correct interactions can be detected in noisy environments. In future research, we will apply various optimizers to optimize the LMF, and the more powerful optimizers are expected to search for more interactions in noisy environments.

Interactions' identification in noisy environments

Although it is a difficult task to detect interactions in noisy environments, it is necessary to develop an effective interaction identification method to form sub-problems by a proper strategy. Explicit averaging [6] can alleviate the uncertainty of noise by re-evaluation. Let the re-evaluating times for $f^N(\mathrm{X})$ be m and $f_i^N(X))$ represents the ith re-evalution value. Then, we apply the principle of Monte Carlo integration [58], and the mean fitness estimation $\bar{f}^N(\mathrm{X})$, standard deviation $\sigma(f^N(X))$, and the standard error of the mean fitness $se(f^N(X))$ are calculated as

$$\bar{f}^N(X) = \frac{1}{m}\sum_{i=1}^{m} f_i^N(X)$$
$$\sigma(f^N(X)) = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m}(f_i^N(X) - \bar{f}^N(X))} \qquad (25)$$
$$se(\bar{f}^N(X)) = \frac{\sigma(f^N(X))}{\sqrt{m}}.$$

Equation (25) shows that sampling an individual's objective function m times can reduce $se(\bar{f}^N(X))$ by a factor of m to improve the accuracy in the mean fitness estimation, which means that the accuracy of sampling increases. It is a feasible method to loosen the threshold ε in DG-based methods and combine the explicit averaging strategy to identify the interactions, although it will consume lots of FEs.

Conclusion

In this paper, we proposed a novel strategy that regards the decomposition problem as a combinatorial optimization problem and designed the LMF to guide the direction of optimization. Besides, we introduce an advanced optimizer named MDE-DS to tackle optimization problems in noisy environments. Numerical experiments show that LMM can detect some interactions in noisy environments, which is competitive with the compared grouping methods. And MDE-DS has a strong ability to search for better solutions, which can accelerate the optimization in noisy environments.

In future research, we will focus on the improvement of LMM and the development of efficient interaction identification methods in noisy environments.

Acknowledgements This work was supported by JSPS KAKENHI under Grant No. JP20K11967.

References

1. Greiner D, Aznarez JJ, Maeso O, Winter G (2010) Single- and multi-objective shape design of y-noise barriers using evolutionary computation and boundary elements. Adv Eng Softw 41(2):368–378. https://doi.org/10.1016/j.advengsoft.2009.06.007
2. Hughes EJ (2001) Evolutionary multi-objective ranking with uncertainty and noise. In: International conference on evolutionary multi-criterion optimization. Springer, pp 329–343. https://doi.org/10.1007/3-540-44719-9_23
3. Li J, Zhou Q, Williams H, Xu H, Du C (2022) Cyber-physical data fusion in surrogate- assisted strength pareto evolutionary algorithm for phev energy management optimization. IEEE Trans Ind Inform 18(6):4107–4117. https://doi.org/10.1109/TII.2021.3121287
4. Sudholt D (2018) On the robustness of evolutionary algorithms to noise: refined results and an example where noise helps. In: Proceedings of the genetic and evolutionary computation conference, pp 1523–1530. https://doi.org/10.1145/3205455.3205595
5. Kim J-S, Jeong U-C, Kim D-W, Han S-Y, Oh J-E (2015) Optimization of sirocco fan blade to reduce noise of air purifier using a metamodel and evolutionary algorithm. Appl Acoust 89:254–266. https://doi.org/10.1016/j.apacoust.2014.10.005
6. Painton L, Diwekar U (1995) Stochastic annealing for synthesis under uncertainty. Eur J Oper Res 83(3):489–502. https://doi.org/10.1016/0377-2217(94)00245-8
7. Diaz J, Handl J (2015) Implicit and explicit averaging strategies for simulation-based optimization of a real-world production planning problem. Informatica (Slovenia) 39:161–168
8. Albukhanajer WA, Briffa JA, Jin Y (2014) Evolutionary multiobjective image feature extraction in the presence of noise. IEEE Trans Cybern 45(9):1757–1768. https://doi.org/10.1109/TCYB.2014.2360074
9. Akimoto Y, Astete-Morales S, Teytaud O (2015) Analysis of runtime of optimization algorithms for noisy functions over discrete codomains. Theor Comput Sci 605:42–50. https://doi.org/10.1016/j.tcs.2015.04.008
10. Chen Y-W, Song Q, Liu X, Sastry PS, Hu X (2020) On robustness of neural architecture search under label noise. Front Big Data. https://doi.org/10.3389/fdata.2020.00002
11. Qian C, Shi J-C, Yu Y, Tang K, Zhou Z-H (2017) Subset selection under noise. Adv Neural Inf Process Syst 30
12. Köppen M (2000) The curse of dimensionality. In: 5th online world conference on soft computing in industrial applications (WSC5), vol 1, pp 4–8
13. Baluja S (1994) Population-based incremental learning. a method for integrating genetic search based function optimization and competitive learning. Technical report, Carnegie-Mellon University, Pittsburgh, Pa, Department of Computer Science
14. Pelikan M, Goldberg DE, Lobo FG (2002) A survey of optimization by building and using probabilistic models. Comput Optim Appl 21(1):5–20. https://doi.org/10.1023/A:1013500812258
15. Moscato P et al. (1989) On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms. Caltech concurrent computation program, C3P report 826, 1989
16. Li E, Wang H, Ye F (2016) Two-level multi-surrogate assisted optimization method for high dimensional nonlinear problems. Appl Soft Comput 46:26–36. https://doi.org/10.1016/j.asoc.2016.04.035
17. Potter MA, De Jong KA (1994) A cooperative coevolutionary approach to function optimization. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics) 866 LNCS, pp 249–257
18. Sun Y, Kirley M, Halgamuge SK (2017) A recursive decomposition method for large scale continuous optimization. IEEE Trans Evolut Comput 22(5):647–661. https://doi.org/10.1109/TEVC.2017.2778089
19. Mei Y, Li X, Yao X (2014) Cooperative coevolution with route distance grouping for large-scale capacitated arc routing problems. IEEE Trans Evolut Comput 18(3):435–449. https://doi.org/10.1109/TEVC.2013.2281503
20. Sayed E, Essam D, Sarker R, Elsayed S (2015) Decomposition-based evolutionary algorithm for large scale constrained problems. Inf Sci 316:457–486. https://doi.org/10.1016/j.ins.2014.10.035. (**nature-inspired algorithms for large scale global optimization**)
21. Omidvar MN, Li X, Yao X (2021) A review of population-based metaheuristics for large-scale black-box global optimization: part a. IEEE Trans Evolut Comput. https://doi.org/10.1109/TEVC.2021.3130838
22. Nabi Omidvar Mohammad, Xiaodong Li, Xin Yao (2021) A review of population-based metaheuristics for large-scale black-box global optimization: part b. IEEE Trans Evolut Comput. https://doi.org/10.1109/TEVC.2021.3130835
23. Munetomo M, Goldberg DE (1999) Linkage identification by non-monotonicity detection for overlapping functions. Evol Comput 7(4):377–398. https://doi.org/10.1162/evco.1999.7.4.377
24. Omidvar MN, Li X, Mei Y, Yao X (2014) Cooperative co-evolution with differential grouping for large scale optimization. IEEE Trans Evolut Comput 18(3):378–393. https://doi.org/10.1109/TEVC.2013.2281543
25. Sun Y, Kirley M, Halgamuge SK (2015) Extended differential grouping for large scale global optimization with direct and indirect variable interactions. In: Proceedings of the 2015 annual conference on genetic and evolutionary computation. GECCO '15. Association for Computing Machinery, New York, NY, USA, pp 313–320. https://doi.org/10.1145/2739480.27546661
26. Omidvar MN, Yang M, Mei Y, Li X, Yao X (2017) DG2: a faster and more accurate differential grouping for large-scale black-box optimization. IEEE Trans Evolut Comput 21(6):929–942. https://doi.org/10.1109/TEVC.2017.2694221

27. Mei Y, Omidvar MN, Li X, Yao X (2016) A competitive divide-and-conquer algorithm for unconstrained large-scale black-box optimization. ACM Trans Math Softw. https://doi.org/10.1145/2791291
28. Yang M, Zhou A, Li C, Yao X (2021) An efficient recursive differential grouping for large-scale continuous problems. IEEE Trans Evolut Comput 25(1):159–171. https://doi.org/10.1109/TEVC.2020.3009390
29. Li X, Tang K, Omidvar MN, Yang Z, Qin K, China H (2013) Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization. Gene 7(33):8
30. Tezuka M, Munetomo M, Akama K (2004) Linkage identification by nonlinearity check for real-coded genetic algorithms. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), vol 3103, pp 222–233
31. van den Bergh F, Engelbrecht AP (2004) A cooperative approach to particle swarm optimization. IEEE Trans Evolut Comput 8(3):225–239. https://doi.org/10.1109/TEVC.2004.826069
32. Tang R-L, Wu Z, Fang Y-J (2017) Adaptive multi-context cooperatively coevolving particle swarm optimization for large-scale problems. Soft Comput 21(16):4735–4754. https://doi.org/10.1007/s00500-016-2081-6
33. Holmstrom L, Koistinen P (1992) Using additive noise in back-propagation training. IEEE Trans Neural Netw 3(1):24–38. https://doi.org/10.1109/72.105415
34. Sancho JM, Miguel MS, Katz SL, Gunton JD (1982) Analytical and numerical studies of multiplicative noise. Phys Rev A 26:1589–1609. https://doi.org/10.1103/PhysRevA.26.1589
35. Jin Y, Branke J (2005) Evolutionary optimization in uncertain environments—a survey. IEEE Trans Evolut Comput 9(3):303–317. https://doi.org/10.1109/TEVC.2005.846356
36. Fitzpatrick JM, Grefenstette JJ (1988) Genetic algorithms in noisy environments. Mach Learn 3(2):101–120
37. Miller BL, Goldberg DE (1996) Genetic algorithms, selection schemes, and the varying effects of noise. Evolut Comput 4(2):113–131. https://doi.org/10.1162/evco.1996.4.2.113
38. Sano Y, Kita H, Kamihira I, Yamaguchi M (2000) Online optimization of an engine controller by means of a genetic algorithm using history of search. In: 2000 26th annual conference of the IEEE Industrial Electronics Society. IECON 2000. 2000 IEEE International conference on industrial electronics, control and instrumentation. 21st century technologies, vol 4, pp 2929–29344. https://doi.org/10.1109/IECON.2000.972463
39. Iacca G, Neri F, Mininno E (2012) Noise analysis compact differential evolution. Int J Syst Sci IJSySc 43:1248–1267. https://doi.org/10.1080/00207721.2011.598964
40. Mininno E, Neri F (2010) A memetic differential evolution approach in noisy optimization. Memet Comput 2:111–135. https://doi.org/10.1007/s12293-009-0029-4
41. Storn R (1996) On the usage of differential evolution for function optimization. In: Proceedings of North American fuzzy information processing, pp 519–523. https://doi.org/10.1109/NAFIPS.1996.534789
42. He X, Zhang Q, Sun N, Dong Y (2009) Feature selection with discrete binary differential evolution. In: 2009 international conference on artificial intelligence and computational intelligence, vol 4, pp 327–330. https://doi.org/10.1109/AICI.2009.438
43. Du J-X, Huang D-S, Wang X-F, Gu X (2007) Shape recognition based on neural networks trained by differential evolution algorithm. Neurocomputing 70(4):896–903. https://doi.org/10.1016/j.neucom.2006.10.026. (**advanced neurocomputing theory and methodology**)
44. Slowik A, Bialko M (2008) Training of artificial neural networks using differential evolution algorithm. In: 2008 conference on human system interactions, pp 60–65. https://doi.org/10.1109/HSI.2008.4581409
45. Ghosh A, Das S, Mallipeddi R, Das AK, Dash SS (2017) A modified differential evolution with distance-based selection for continuous optimization in presence of noise. IEEE Access 5:26944–26964. https://doi.org/10.1109/ACCESS.2017.2773825
46. Ghosh A, Das S, Mullick SS, Mallipeddi R, Das AK (2017) A switched parameter differential evolution with optional blending crossover for scalable numerical optimization. Appl Soft Comput 57:329–352. https://doi.org/10.1016/j.asoc.2017.03.003
47. Kundu R, Mukherjee R, Das S, Vasilakos AV (2013) Adaptive differential evolution with difference mean based perturbation for dynamic economic dispatch problem. In: 2013 IEEE symposium on differential evolution (SDE), pp 38–45. https://doi.org/10.1109/SDE.2013.6601440
48. Sun Y, Kirley M, Halgamuge SK (2018) A recursive decomposition method for large scale continuous optimization. IEEE Trans Evolut Comput 22(5):647–661. https://doi.org/10.1109/TEVC.2017.2778089
49. Sun Y, Omidvar MN, Kirley M, Li X (2018) Adaptive threshold parameter estimation with recursive differential grouping for problem decomposition. In: Proceedings of the genetic and evolutionary computation conference. GECCO '18. Association for Computing Machinery, New York, NY, USA, pp 889–896. https://doi.org/10.1145/3205455.3205483
50. Wu Y, Peng X, Wang H, Jin Y, Xu D (2022) Cooperative coevolutionary CMA-ES with landscape-aware grouping in noisy environments. IEEE Trans Evolut Comput. https://doi.org/10.1109/TEVC.2022.3180224
51. Sun Y, Kirley M, Halgamuge SK (2015) Extended differential grouping for large scale global optimization with direct and indirect variable interactions. In: Proceedings of the 2015 annual conference on genetic and evolutionary computation. GECCO '15. Association for Computing Machinery, New York, NY, USA, pp 313–320. https://doi.org/10.1145/2739480.2754666
52. Munetomo M (2002) Linkage identification with epistasis measures considering monotonicity conditions. In: Proceedings of the 4th Asia-Pacific conference on simulated evolution and learning. https://cir.nii.ac.jp/crid/1570009750206806528
53. Zhong R, Munetomo M (2022) Random population-based decomposition method by linkage identification with non-linearity minimization on graph. In: Transactions on computational science and computational intelligence. Springer
54. De Jong KA (1975) An analysis of the behavior of a class of genetic adaptive systems. Ph.D. thesis, University of Michigan, USA. AAI7609381
55. Omidvar MN, Li X, Yao X (2010) Cooperative co-evolution with delta grouping for large scale non-separable function optimization, pp 1–8. https://doi.org/10.1109/CEC.2010.5585979
56. Yang Z, Tang K, Yao X (2008) Large scale evolutionary optimization using cooperative coevolution. Inf Sci 178(15):2985–2999. https://doi.org/10.1016/j.ins.2008.02.017. (**nature inspired problem-solving**)
57. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evolut Comput 1(1):67–82. https://doi.org/10.1109/4235.585893
58. Gopalakrishnan G, Minsker BS, Goldberg DE (2001) Optimal sampling in a noisy genetic algorithm for risk-based remediation design. J Hydroinform 5:11–25. https://doi.org/10.1061/40569(2001)94