Adaptive Threshold Parameter Estimation with Recursive Differential Grouping for Problem Decomposition

Yuan Sun
School of Computing and Information Systems
The University of Melbourne
Parkville, VIC, Australia
yuan.sun@unimelb.edu.au

Mohammad Nabi Omidvar
School of Computer Science
The University of Birmingham
Birmingham, United Kingdom
m.omidvar@cs.bham.ac.uk

Michael Kirley
School of Computing and Information Systems
The University of Melbourne
Parkville, VIC, Australia
mkirley@unimelb.edu.au

Xiaodong Li
School of Science
RMIT University
Melbourne, VIC, Australia
xiaodong.li@rmit.edu.au

ABSTRACT

Problem decomposition plays an essential role in the success of cooperative co-evolution (CC), when used for solving large-scale optimization problems. The recently proposed *recursive differential grouping* (RDG) method has been shown to be very efficient, especially in terms of time complexity. However, it requires an appropriate parameter setting to estimate a threshold value in order to determine if two subsets of decision variables interact or not. Furthermore, using one global threshold value may be insufficient to identify variable interactions in components with different contribution to the fitness value. Inspired by the *different grouping 2* (DG2) method, in this paper, we adaptively estimates a threshold value based on computational round-off errors for RDG. We derive an upper bound of the round-off errors, which is shown to be sufficient when identifying variable interactions across a wide range of large-scale benchmark problems. Comprehensive numerical experimental results showed that the proposed RDG2 method achieved higher decomposition accuracy than RDG and DG2. When embedded into a CC framework, it achieved statistically equal or significantly better solution quality than RDG and DG2, when used to solve the benchmark problems.

CCS CONCEPTS

• **Theory of computation → Non-parametric optimization**; **Divide and conquer**;

KEYWORDS

Large-scale continuous optimization, cooperative co-evolution, problem decomposition, parameter adaptation, round-off error analysis

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5618-3/18/07…$15.00
https://doi.org/10.1145/3205455.3205483

ACM Reference Format:
Yuan Sun, Mohammad Nabi Omidvar, Michael Kirley, and Xiaodong Li. 2018. Adaptive Threshold Parameter Estimation with Recursive Differential Grouping for Problem Decomposition. In *GECCO '18: Genetic and Evolutionary Computation Conference, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205483

1 INTRODUCTION

The cooperative co-evolution (CC) [15] framework has been applied with some success when scaling up evolutionary algorithms to solve high-dimensional (large-scale) optimization problems [10, 12, 16]. It divides a large-scale optimization problem into a number of low-dimensional components that are solved cooperatively. The main challenge when using the CC framework lies in problem decomposition, which is the process of identifying and grouping interacting decision variables into respective components [2, 9, 12]. A good decomposition should reflect the underlying interaction structure of the decision variables [12, 14, 20].

The recently proposed recursive differential grouping (RDG) [20] method achieves high computational efficiency by recursively examining the interaction between two subsets of decision variables (instead of two variables commonly used in most decomposition algorithms). The number of function evaluations (FEs) used by RDG to decompose an n-dimensional problem has been shown to be less than $6n\log(n)$. The RDG method approximates a global threshold value based on the magnitude of the objective value, which is then used to identify variable interactions in a given problem. RDG requires users to specify an appropriate parameter value to estimate the threshold, however this parameter is highly dependent on the structural property of the problem. Further, using only one threshold value may be insufficient to completely identify variable interactions in a given problem with imbalanced components [8, 14].

In this paper, we introduce an adaptive threshold value estimation mechanism for RDG, which is inspired by the differential grouping 2 (DG2) [12] method. We derive an upper bound for the computational round-off error incurred by the floating-point operations, which is then used as the threshold value to differentiate between the separable and non-separable decision variables. The threshold value is estimated adaptively without the need for parameter setting. For any two subsets of the decision variables, a

different threshold value is estimated to identify the interaction between the two subsets.

We evaluated the proposed RDG2 method (RDG with parameter adaptation) using the CEC'2010 [22] and CEC'2013 [8] large-scale global optimization benchmark problems. Comprehensive numerical experiments confirmed the effectiveness of RDG2: 1) the decomposition accuracy generated by RDG2 was equal to or higher than that generated by both DG2 and RDG; and 2) when embedded into a CC framework to solve the benchmark problems, RDG2 achieved statistically similar or significantly better solutions than the other two decomposition methods.

The remainder of this paper is organized as follows. In Section 2, we briefly review the existing decomposition methods in the literature. In Section 3, the threshold value adaptation mechanism for RDG2 is described in detail. Section 4 describes the experimental methodology to evaluate the proposed RDG2 method, and analyzes the experimental results. The final section concludes the paper and suggests possible future research directions.

2 RELATED WORK

The existing decomposition methods can be classified into two different approaches: the *blind decomposition* method (e.g., univariable grouping [15], S_k grouping [24] and random grouping [25]) does not take the underlying structure of variable interactions into consideration. The formal definition of pairwise variable interaction is described as follows.

Definition 2.1. (Sun et al. [20, 21]) Let $f : \mathbb{R}^n \rightarrow \bar{\mathbb{R}}$ be a differentiable function. Decision variables x_i and x_j interact if a candidate solution $\mathbf{x}^*$ exists, such that

$$\frac{\partial^2 f(\mathbf{x}^*)}{\partial x_i \partial x_j} \neq 0, \tag{1}$$

which is denoted by $x_i \leftrightarrow x_j$. Decision variables x_i and x_j conditionally interact if for any candidate solution $\mathbf{x}^*$,

$$\frac{\partial^2 f(\mathbf{x}^*)}{\partial x_i \partial x_j} = 0, \tag{2}$$

and a set of decision variables $\{x_{k_1}, \ldots, x_{k_t}\} \subset X$ exists, such that $x_i \leftrightarrow x_{k_1} \leftrightarrow \ldots \leftrightarrow x_{k_t} \leftrightarrow x_j$, where $k_1, \cdots, k_t$ are t decision variable indices. Decision variables x_i and x_j are independent if for any candidate solution $\mathbf{x}^*$, Eq. (2) holds and a set of decision variables $\{x_{k_1}, \ldots, x_{k_t}\} \subset X$ does not exist, such that $x_i \leftrightarrow x_{k_1} \leftrightarrow \ldots \leftrightarrow x_{k_t} \leftrightarrow x_j$.[1]

In the second method, *intelligent decomposition*, the structure of components is determined by the identified variable interactions [3, 12, 14, 20]. Take a 6-dimensional problem as an example, where x_3 interacts with x_2 and x_4, and x_5 interacts with x_6. The problem can be decomposed into three components $\{x_1\}$, $\{x_2, x_3, x_4\}$, $\{x_5, x_6\}$, as shown in Figure 1. Therefore, it is of vital importance for an intelligent decomposition method to accurately and efficiently identify variable interactions.[2]

[1] In the following, variable interaction refers to the "interaction" (instead of "conditional interaction") between decision variables.

[2] The terminologies "blind decomposition" and "intelligent decomposition" are suggested by Xiaodong Li, which are more appropriate than "manual decomposition" and "automatic decomposition" we used in [20].

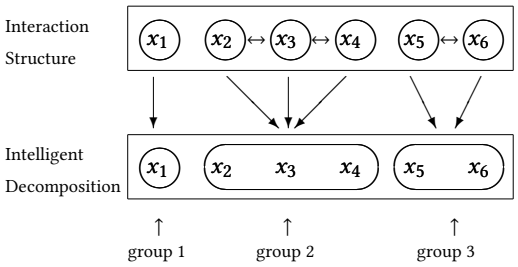

Figure 1: The variable interaction structure and intelligent decomposition. The notation $x_i \leftrightarrow x_j$ denotes that decision variable x_i interacts with x_j.

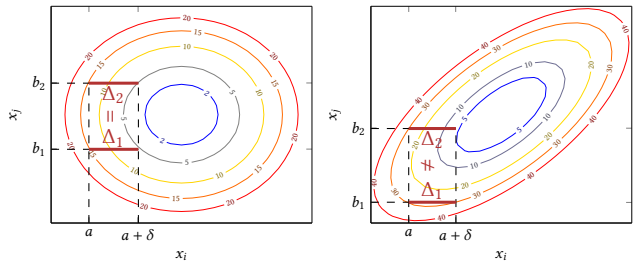

Figure 2: The rationale behind the non-linearity detection method when identifying (a) separable and (b) non-separable decision variables. In the separable contour plot (a), the fitness change induced by adding a perturbation δ to the decision variable x_i is the same for different values of x_j. However in the non-separable contour plot (b), the fitness change induced by perturbing x_i varies for different values of x_j.

The non-linearity detection [12, 23] method identifies variable interactions by detecting the fitness changes when perturbing the decision variables. If the fitness change induced by perturbing decision variable x_i varies for different values of x_j, then one concludes that x_i and x_j interact. The rationale behind the non-linearity detection method is shown in Theorem 2.2 and Figure 2.

THEOREM 2.2. *(Omidvar et al. [12]) Let $f : \mathbb{R}^n \rightarrow \bar{\mathbb{R}}$ be an objective function. Decision variable x_i interacts with x_j, if there exist real numbers a, b_1, b_2 and $\delta \neq 0$ such that*

$$\Delta_{\delta,x_i}[f](\mathbf{x})|_{x_i=a,x_j=b_1} \neq \Delta_{\delta,x_i}[f](\mathbf{x})|_{x_i=a,x_j=b_2}, \tag{3}$$

where

$$\Delta_{\delta,x_i}[f](\mathbf{x}) = f(\cdots, x_i + \delta, \cdots) - f(\cdots, x_i, \cdots). \tag{4}$$

A number of decomposition methods have been proposed based on non-linearity detection, e.g., differential grouping [12], extended differential grouping [19], global differential grouping [11], and DG2 [14]. These methods typically check for interactions between pairs of decision variables, requiring $O(n^2)$ FEs to decompose an n-dimensional problem. The fast interdependency identification (FII) [7] method improves the decomposition efficiency by identifying

the interaction between one decision variable with the remaining decision variables. However, the number of FEs used by FII is still in $O(n^2)$ when decomposing overlapping problems (e.g., the Rosenbrock function [8]).

The recently proposed RDG [20] method has been shown to be able to decompose any n-dimensional problem using $O(n\log(n))$ FEs. It identifies the interaction between two subsets of decision variables based on Theorem 2.3.

Notation 1. *Let X be the set of decision variables $\{x_1,\ldots,x_n\}$; U_X be the set of unit vectors in the decision space R^n. Let X_1 be a subset of decision variables $X_1 \subset X$; and U_{X_1} be a subset of U_X such that any unit vector $\mathbf{u}=(u_1,\ldots,u_n) \in U_{X_1}$, we have*

$$u_i = 0, \qquad \text{if } x_i \notin X_1. \tag{5}$$

Theorem 2.3. *(Sun et al. [20]) Let $f\colon \mathbb{R}^n \to \bar{\mathbb{R}}$ be an objective function; $X_1 \subset X$ and $X_2 \subset X$ be two mutually exclusive subsets of decision variables: $X_1 \cap X_2 = \emptyset$. If there exist 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 candidate solution $\mathbf{x}^*$ in the decision space, such that*

$$f(\mathbf{x}^* + l_1\mathbf{u}_1 + l_2\mathbf{u}_2) - f(\mathbf{x}^* + l_2\mathbf{u}_2) \neq f(\mathbf{x}^* + l_1\mathbf{u}_1) - f(\mathbf{x}^*), \tag{6}$$

there is some interaction between decision variables in X_1 and X_2.

The RDG method identifies the interaction between two subsets of decision variables (X_1 and X_2) using the following procedure:

(1) Set all the decision variables to the lower bounds (**lb**) of the search space ($\mathbf{x}_{l,l}$);
(2) Perturb the decision variables X_1 of $\mathbf{x}_{l,l}$ from the lower bounds to the upper bounds (**ub**), denoted by $\mathbf{x}_{u,l}$;
(3) Calculate the fitness difference (Δ_1) between $\mathbf{x}_{l,l}$ and $\mathbf{x}_{u,l}$;
(4) Perturb the decision variables X_2 of $\mathbf{x}_{l,l}$ and $\mathbf{x}_{u,l}$ from the lower bounds to the middle between the lower bounds and upper bounds, denoted by $\mathbf{x}_{l,m}$ and $\mathbf{x}_{u,m}$ respectively;
(5) Calculate the fitness difference (Δ_2) between $\mathbf{x}_{l,m}$ and $\mathbf{x}_{u,m}$;
(6) If the difference (λ) between Δ_1 and Δ_2 is greater than a threshold ϵ, there is some interaction between X_1 and X_2.

The two subscripts of $\mathbf{x}$ denote the values of X_1 and X_2 respectively: 'l' is lower bound; 'u' is upper bound; and 'm' is the mean (middle) of the lower and upper bounds. Based on this, a recursive grouping procedure is used to decompose a problem (see supplementary material or [20] for details).[3]

In theory, the threshold ϵ can be set to zero, as any positive value of the non-linearity term ($\lambda = |\Delta_1 - \Delta_2|$) implies an interaction between the subset of decision variables under examination. However in practice, the value of λ for separable decision variables may be non-zero, due to the computational round-off errors incurred by the floating-point operations (see Section 3 for details). Therefore, a positive threshold value ($\epsilon > 0$) is required to differentiate the genuine non-zero λ values.

The RDG method estimates a threshold value based on the magnitude of the objective value [11]:

$$\epsilon := \alpha \cdot \min\left\{|f(\mathbf{x}_1)|, \cdots, |f(\mathbf{x}_k)|\right\}, \tag{7}$$

where $\mathbf{x}_1, \cdots, \mathbf{x}_k$ are k randomly generated candidate solutions, and α is the control coefficient [11]. However, it is non-trivial to select an appropriate value for α. Moreover, RDG employs a global threshold value to identify the interaction between all pairs of decision variable subsets in a given problem, which may be insufficient to deal with problems with imbalanced components [8, 14].

The DG2 [14] method addresses these issues by automatically estimating a "greatest lower bound" (e_{inf}) and a "least upper bound" (e_{sup}) of the round-off errors involved in calculating the non-linearity term (λ). The decision variables are regarded as separable if $\lambda < e_{\text{inf}}$, and non-separable if $\lambda > e_{\text{sup}}$. If λ falls between e_{inf} and e_{sup}, the threshold value is set to a weighted average of e_{inf} and e_{sup}. The weight is calculated as the relative proportion of the separable and non-separable decision variable pairs that have been identified.

3 THRESHOLD PARAMETER ADAPTATION FOR RDG

In this section, we derive an upper bound of the round-off errors incurred by the calculation of the non-linearity term λ, which is then used as the threshold value for RDG to identify variable interactions.

The arithmetic performed in a modern computing device operates on floating-point instead of real numbers, which generates two types of round-off errors: the representation error and arithmetic error. The representation error results from rounding a real number to the nearest floating-point number. Let $fl : \mathbb{R} \to \mathbb{F}$ denote a mapping from the set of real numbers ($\mathbb{R}$) to the set of floating-point numbers ($\mathbb{F}$). The relative representation error (δ) of a real number x is defined as

$$\delta = \frac{fl(x) - x}{x}. \tag{8}$$

According to the IEEE 754 standard [1], the absolute value of the relative representation error (δ) is bounded by a machine dependent constant μ_{M}, which is half of the machine epsilon ϵ_{M}: $|\delta| < \mu_{\text{M}}$.[4] Therefore, the absolute value of the absolute representation error ($|\delta x|$) may grow with the magnitude of x.

The arithmetic error comes from the floating-point arithmetic, for example the floating-point summation ($\oplus$).[5] The IEEE 754 standard guarantees that the floating-point sum of two real numbers is equal to the floating-point number closest to the real sum of the two numbers: $x_1 \oplus x_2 = fl(x_1 + x_2)$. However, this statement can not be generalized to a series of floating-point sums due to the accumulation of round-off errors. In other words, there is no guarantee that the equality $x_1 \oplus x_2 \oplus \cdots \oplus x_n = fl(x_1 + x_2 + \cdots + x_n)$ is true for $n \geq 3$. This property can be generalized to floating-point subtraction, multiplication and division.

The round-off error involved in the calculation of the non-linearity term $\lambda = \left|\left(f(\mathbf{x}_{l,l}) - f(\mathbf{x}_{u,l})\right) - \left(f(\mathbf{x}_{l,m}) - f(\mathbf{x}_{u,m})\right)\right|$ comes from two sources: 1) the arithmetic floating-point subtraction between the fitness values $f(\mathbf{x})$, and 2) the calculation of the fitness values $f(\mathbf{x})$. We first calculate the round-off error resulted from the arithmetic floating-point subtraction between the fitness values. As the IEEE 754 standard guarantees that: $x_1 \ominus x_2 = fl(x_1 - x_2)$, we have

$$\hat{\Delta}_1 = \hat{f}(\mathbf{x}_{l,l}) \ominus \hat{f}(\mathbf{x}_{u,l}) = \left(\hat{f}(\mathbf{x}_{l,l}) - \hat{f}(\mathbf{x}_{u,l})\right)(1 + \delta_1), \tag{9}$$

where $|\delta_1| < \mu_{\text{M}}$ is the relative representation error. We use $\hat{x}$ to denote the floating-point number of x for the sake of simplicity.

[3] The supplementary material is available at https://doi.org/10.1145/3205455.3205483.

[4] In the numerical computing software MATLAB, $\epsilon_{\text{M}} = 2^{-52}$ and $\mu_{\text{M}} = 2^{-53}$.

[5] Circled arithmetic operators (e.g., $\oplus$, $\ominus$, $\otimes$ and $\oslash$) denote floating-point operators.

Similarly,

$$\hat{\Delta}_2 = \hat{f}(\mathbf{x}_{l,m}) \ominus \hat{f}(\mathbf{x}_{u,m}) = \big(\hat{f}(\mathbf{x}_{l,m}) - \hat{f}(\mathbf{x}_{u,m})\big)(1+\delta_2). \quad (10)$$

Thus,

$$\begin{aligned}\hat{\lambda} &= \big|\hat{\Delta}_1 \ominus \hat{\Delta}_2\big| = \big|(\hat{\Delta}_1 - \hat{\Delta}_2)(1+\delta_3)\big| \\ &= \big|\big(\hat{f}(\mathbf{x}_{l,l}) - \hat{f}(\mathbf{x}_{u,l})\big)(1+\delta_1)(1+\delta_3) \\ &\quad - \big(\hat{f}(\mathbf{x}_{l,m}) - \hat{f}(\mathbf{x}_{u,m})\big)(1+\delta_2)(1+\delta_3)\big|.\end{aligned} \quad (11)$$

In order to find an upper bound on the accumulated arithmetic error, the following theorem is used.

Theorem 3.1. *(Corless and Fillion [4]) Given a floating-point number system that satisfies IEEE 754 Standard [1] such that $|\delta_i| < \mu_{\mathrm{M}}$, and $k\mu_{\mathrm{M}} < 1$, we have:*

$$\prod_{i=1}^{k}(1+\delta_i)^{e_i} = 1+\theta_k, \quad (12)$$

where

$$|\theta_k| \le \frac{k\mu_{\mathrm{M}}}{1-k\mu_{\mathrm{M}}} := \gamma_k,\ e_i = \pm 1. \quad (13)$$

Theorem 3.1 states that the product $\prod_{i=1}^{k}(1+\delta_i)^{\pm 1}$ can be written as $(1+\theta_k)$, where $|\theta_k|$ is bounded by a machine dependent constant γ_k, which is defined as $k\mu_{\mathrm{M}}/(1-k\mu_{\mathrm{M}})$. In Eq. (11), $k=2$ and

$$\hat{\lambda} = \big|\big(\hat{f}(\mathbf{x}_{l,l}) - \hat{f}(\mathbf{x}_{u,l})\big)(1+\theta_2) - \big(\hat{f}(\mathbf{x}_{l,m}) - \hat{f}(\mathbf{x}_{u,m})\big)(1+\theta_2')\big|, \quad (14)$$

where $|\theta_2| \le \gamma_2$ and $|\theta_2'| \le \gamma_2$.

In the next step, we estimate the round-off error associated with the calculation of the fitness value $f(\mathbf{x})$. As the objective functions are "black-box", we do not know the exact order of $(1+\delta_i)$. To overcome this difficulty, we introduce the following assumptions.

Assumption 3.2. *(Higham [6]) The round-off error grows with the square root of the number of floating-point operations (Φ) involved in a calculation.*

In other words, to calculate an upper bound on the round-off error involved in the calculation of the fitness value based on Theorem 3.1, we assume $k \approx \sqrt{\Phi}$. However, in black-box optimization, the number of floating-point operations involved in the calculation of the objective function is also unknown. To overcome this difficulty, we assume that the number of floating-point operations (Φ) has a linear relationship with the dimensionality of the problem (n) [14].

Assumption 3.3. *(Omidvar et. al. [14]) The number of floating-point operations (Φ) involved in the calculation of a black-box objective function is in the order of $\Theta(n)$, where n is the dimensionality of the objective function.*

This linear assumption is a safe choice as 1) most polynomial evaluations fall into this group, 2) the upper bound calculated based on Theorem 3.1 is very conservative; the actual round-off errors are much smaller in practice [18], and 3) over-estimating the threshold value is detrimental to the detection of interacting decision variables [14]. Additionally, the empirical evidence in [14] show that the linear assumption is more reliable than quadratic or cubic assumptions. Therefore, we use linear assumption, and specifically let $\Phi \approx n$.

Theorem 3.4. *Under Assumption 3.2 and Assumption 3.3, an upper bound on the round-off errors associated with the calculation of the non-linearity term λ is given by*

$$|\lambda - \hat{\lambda}| \le \gamma_{\sqrt{n}+2}\big(|f(\mathbf{x}_{l,l})| + |f(\mathbf{x}_{u,l})| + |f(\mathbf{x}_{l,m})| + |f(\mathbf{x}_{u,m})|\big). \quad (15)$$

Proof. Under Assumption 3.2 and Assumption 3.3, we have $k = \sqrt{n}$, and

$$\hat{f}(\mathbf{x}) = (1+\theta_{\sqrt{n}})f(\mathbf{x}). \quad (16)$$

Substituting Eq. (16) into Eq. (14),

$$\begin{aligned}\hat{\lambda} = \big|&f(\mathbf{x}_{l,l})(1+\theta_{\sqrt{n}})(1+\theta_2) - f(\mathbf{x}_{u,l})(1+\theta'_{\sqrt{n}})(1+\theta_2) \\ &- f(\mathbf{x}_{l,m})(1+\tilde{\theta}_{\sqrt{n}})(1+\theta_2') + f(\mathbf{x}_{u,m})(1+\bar{\theta}_{\sqrt{n}})(1+\theta_2')\big|,\end{aligned} \quad (17)$$

where $|\theta_{\sqrt{n}}|$, $|\theta'_{\sqrt{n}}|$, $|\tilde{\theta}_{\sqrt{n}}|$ and $|\bar{\theta}_{\sqrt{n}}|$ are bounded by $\gamma_{\sqrt{n}}$. In Theorem 3.1, it is true that $(1+\theta_i)(1+\theta_j) = (1+\theta_{i+j})$. Therefore,

$$\begin{aligned}\hat{\lambda} = \big|&f(\mathbf{x}_{l,l})(1+\theta_{\sqrt{n}+2}) - f(\mathbf{x}_{u,l})(1+\theta'_{\sqrt{n}+2}) \\ &- f(\mathbf{x}_{l,m})(1+\tilde{\theta}_{\sqrt{n}+2}) + f(\mathbf{x}_{u,m})(1+\bar{\theta}_{\sqrt{n}+2})\big|.\end{aligned} \quad (18)$$

As the inequality $|x_1 + x_2| \le |x_1| + |x_2|$ holds for any real numbers x_1 and x_2,

$$\begin{aligned}\hat{\lambda} \le \big|&f(\mathbf{x}_{l,l}) - f(\mathbf{x}_{u,l}) - f(\mathbf{x}_{l,m}) + f(\mathbf{x}_{u,m})\big| + \big|f(\mathbf{x}_{l,l})\theta_{\sqrt{n}+2} \\ &- f(\mathbf{x}_{u,l})\theta'_{\sqrt{n}+2} - f(\mathbf{x}_{l,m})\tilde{\theta}_{\sqrt{n}+2} + f(\mathbf{x}_{u,m})\bar{\theta}_{\sqrt{n}+2}\big|.\end{aligned} \quad (19)$$

As the inequality $|\sum_{i=1}^{m} x_i| \le \sum_{i=1}^{m} |x_i|$ holds for any positive integer m,

$$\begin{aligned}\hat{\lambda} &\le \lambda + \big|f(\mathbf{x}_{l,l})\theta_{\sqrt{n}+2}\big| + \big|f(\mathbf{x}_{u,l})\theta'_{\sqrt{n}+2}\big| \\ &\quad + \big|f(\mathbf{x}_{l,m})\tilde{\theta}_{\sqrt{n}+2}\big| + \big|f(\mathbf{x}_{u,m})\bar{\theta}_{\sqrt{n}+2}\big| \\ &\le \lambda + \gamma_{\sqrt{n}+2}\big(|f(\mathbf{x}_{l,l})| + |f(\mathbf{x}_{u,l})| + |f(\mathbf{x}_{l,m})| + |f(\mathbf{x}_{u,m})|\big).\end{aligned} \quad (20)$$

Therefore,

$$\hat{\lambda} - \lambda \le \gamma_{\sqrt{n}+2}\big(|f(\mathbf{x}_{l,l})| + |f(\mathbf{x}_{u,l})| + |f(\mathbf{x}_{l,m})| + |f(\mathbf{x}_{u,m})|\big). \quad (21)$$

On the other hand, as the inequality $|x_1 + x_2| \ge |x_1| - |x_2|$ holds for any real numbers x_1 and x_2, similarly we can obtain

$$\lambda - \hat{\lambda} \le \gamma_{\sqrt{n}+2}\big(|f(\mathbf{x}_{l,l})| + |f(\mathbf{x}_{u,l})| + |f(\mathbf{x}_{l,m})| + |f(\mathbf{x}_{u,m})|\big). \quad (22)$$

Thus,

$$|\lambda - \hat{\lambda}| \le \gamma_{\sqrt{n}+2}\big(|f(\mathbf{x}_{l,l})| + |f(\mathbf{x}_{u,l})| + |f(\mathbf{x}_{l,m})| + |f(\mathbf{x}_{u,m})|\big). \quad (23)$$

□

The upper bound of round-off errors is then used as the threshold value (ϵ) to distinguish between separable and non-separable decision variable subsets:

$$\epsilon := \gamma_{\sqrt{n}+2}\big(|f(\mathbf{x}_{l,l})| + |f(\mathbf{x}_{u,l})| + |f(\mathbf{x}_{l,m})| + |f(\mathbf{x}_{u,m})|\big). \quad (24)$$

Decision variable subsets are regarded as interacting if $\hat{\lambda} > \epsilon$, and separable if $\hat{\lambda} \le \epsilon$.

4 EXPERIMENTS

4.1 Methodology

To evaluate the efficacy of the proposed RDG2 method, two research questions guide the experimental study:

Q1. Can the proposed RDG2 method be used to accurately decompose large-scale optimization problems?
Q2. Can the proposed RDG2 method generate good solution quality when embedded into a CC framework to solve large-scale optimization problems?

To answer Q1, the proposed RDG2 method was used to decompose the CEC'2010 [22] and CEC'2013 [8] benchmark problems.[6] Two metrics were employed to evaluate the performance of the RDG2 method: 1) the number of FEs used to decompose a problem; and 2) the percentage of interacting decision variables that are correctly grouped [20]. The performance of the RDG2 method was then compared against the performance of the RDG (with $\alpha = 10^{-12}$) [20] and DG2 [14] methods. Note that RDG2 and DG2 are parameter-free.

To answer Q2, the proposed RDG2 method was embedded into a CC [11] framework to solve the CEC'2010 and CEC'2013 benchmark problems. This CC framework used the well-performed Covariance Matrix Adaptation - Evolutionary Strategy (CMA-ES) [5] algorithm to solve each component. The parameter settings for CMA-ES were consistent with the original paper. The maximum number of FEs was set to 3×10^6, divided between the decomposition stage and optimization stage. For each benchmark problem, the median, mean and standard deviation of the best solutions found by the CC-RDG2 algorithm (with CMA-ES as the component optimizer) based on 25 independent runs were recorded. The performance of the RDG2 method was compared against the performance of the RDG and DG2 methods, when embedded in the CC framework.[7]

The Kruskal-Wallis nonparametric one-way ANOVA test [17] with 95% confidence interval was used to determine whether the performance of at least one algorithm was significantly different from the others. Then a series of Wilcoxon rank-sum tests (significance level = 0.05) with Holm p-value correction [17] was conducted in a pairwise fashion to find the best performing algorithm(s).

4.2 Experimental Results

The decomposition comparison between RDG2 and the other two methods is presented in Section 4.2.1, while the optimization comparison is presented in Section 4.2.2.

4.2.1 Decomposition Comparison. The experimental results of the RDG2, RDG and DG2 methods when used to decompose the CEC'2013 benchmark problems are shown in Table 1. The RDG2 method consistently generated equally well or better results than RDG and DG2 on the partially separable problems (f_4 to f_{11}).

The first three problems (f_1-f_3) are fully separable. Therefore, decomposition accuracy is not applicable to these problems [20]. The CEC'2013 f_{13} and f_{14} are benchmark problems with overlapping (conforming or conflicting) components. It is not yet clear what is the best approach to decompose these problems [8, 13, 20]. The existing intelligent decomposition methods place all the overlapping components into one group. On the other benchmark problems where the components are independent from each other, the "ideal" decomposition can possibly be achieved [8, 13, 20]. Note that the 100% decomposition accuracy in Table 1 corresponds to the ideal decomposition.

[6]The MATLAB source code of RDG2 is available at https://bitbucket.org/yuans/rdg2.
[7]If not specified, CMA-ES is always used as the component optimizer.

Table 1: The experimental results of the RDG2, RDG (with $\alpha = 10^{-12}$) and DG2 methods when used to decompose the CEC'2013 benchmark problems. "a" denotes the decomposition accuracy; "FEs" denotes the function evaluations used. The entries with higher decomposition accuracy are highlighted in bold. Different categories of benchmark problems are divided by the lines.

Func	RDG2		RDG ($\alpha = 10^{-12}$)		DG2	
ID	a	FEs	a	FEs	a	FEs
f_1	–	2.99e+03	–	3.00e+03	–	5.00e+05
f_2	–	3.04e+03	–	3.00e+03	–	5.00e+05
f_3	–	5.99e+03	–	6.00e+03	–	5.00e+05
f_4	100%	9.83e+03	100%	9.84e+03	100%	5.00e+05
f_5	100%	9.83e+03	100%	1.01e+04	100%	5.00e+05
f_6	100%	1.12e+04	100%	1.32e+04	100%	5.00e+05
f_7	**100%**	9.81e+03	**100%**	9.82e+03	83.3%	5.00e+05
f_8	**80.0%**	1.91e+04	**80.0%**	1.95e+04	78.5%	5.00e+05
f_9	100%	1.91e+04	100%	1.92e+04	100%	5.00e+05
f_{10}	**100%**	1.93e+04	82.7%	1.91e+04	**100%**	5.00e+05
f_{11}	**100%**	1.93e+04	10.0%	1.06e+04	**100%**	5.00e+05
f_{12}	100%	5.08e+04	100%	5.08e+04	100%	5.00e+05
f_{13}	–	1.51e+04	–	8.39e+03	–	4.10e+05
f_{14}	–	1.61e+04	–	1.61e+04	–	4.10e+05
f_{15}	100%	5.99e+03	100%	6.16e+03	100%	5.00e+05

On f_{10} and f_{11}, RDG2 achieved 100% decomposition accuracy, while the decomposition accuracy of RDG was low. On f_7 and f_8, RDG2 generated higher decomposition accuracy than DG2. This observation implied that the threshold value estimated by RDG2 was more reliable than those estimated by RDG and DG2. The DG2 method approximated the "greatest lower bound" (e_{inf}) and "least upper bound" (e_{sup}) of the round-off errors as follows:

$$e_{\text{inf}} := \gamma_2 \max\left\{|f(\mathbf{x}_{l,l})| + |f(\mathbf{x}_{u,m})|, |f(\mathbf{x}_{u,l})| + |f(\mathbf{x}_{l,m})|\right\},$$
$$e_{\text{sup}} := \gamma_{\sqrt{n}} \max\left\{f(\mathbf{x}_{l,l}), f(\mathbf{x}_{u,l}), f(\mathbf{x}_{l,m}), f(\mathbf{x}_{u,m})\right\}.$$

Therefore, the threshold value used by DG2 was smaller than that by RDG2. The RDG method calculated a global threshold value based on Eq. (7).

To gain deeper insight into the performance of RDG2, we select two benchmark problems: f_8 and f_{11}, each of which consists of 20 non-separable components with totally 1000 decision variables. The contribution of each component to the overall fitness value is unbalanced. The weight value of each component is shown in Figure 3a and Figure 3b: higher weight values indicate larger contribution to the fitness value.

We recorded the floating-point value of the non-linearity term when the corresponding decision variable subsets interact (denoted

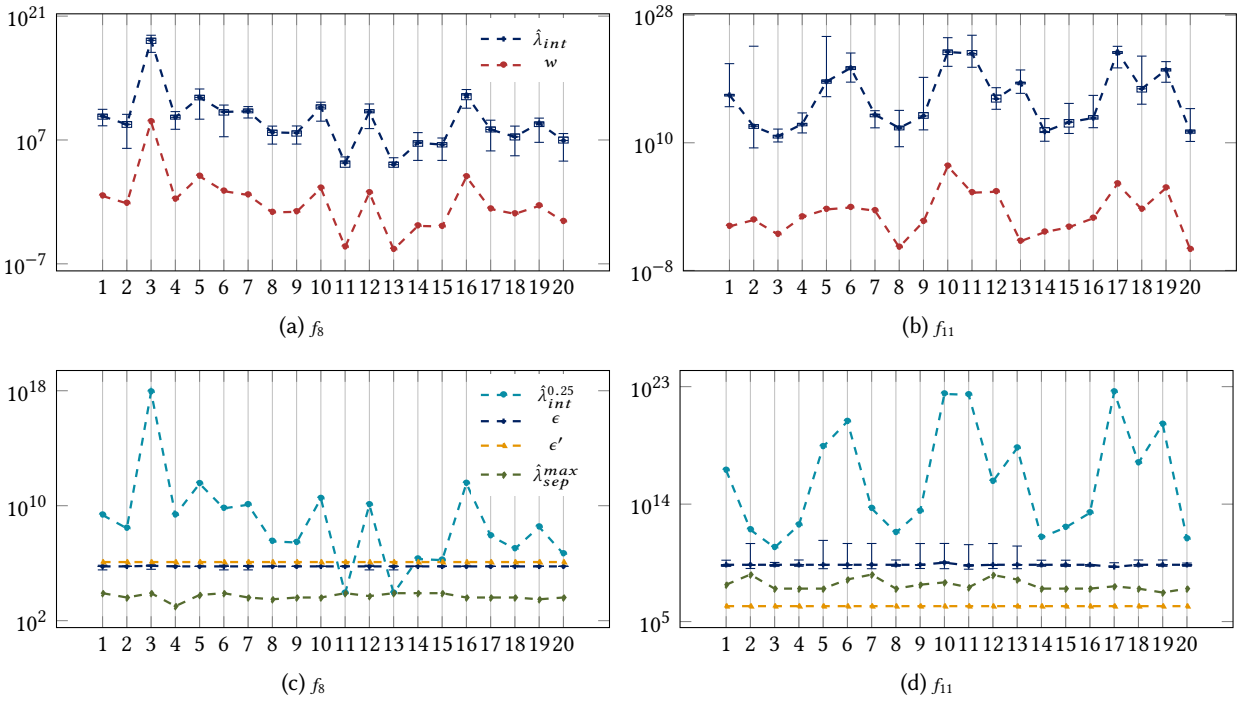

Figure 3: The plots of the threshold value, non-linearity term and weight value of each component on the CEC'2013 f_8 and f_{11}. The horizontal axis represents the indices of each component. In (a) and (b), $\hat{\lambda}_{int}$ denotes the floating-point value of the non-linear term ($\hat{\lambda}$) when the corresponding decision variable subsets interact; "w" denotes the weight value, which represents the contribution of each component to the overall fitness value. In (c) and (d), $\hat{\lambda}^{0.25}_{int}$ denotes the 25% percentile of the $\hat{\lambda}_{int}$ values; ϵ and ϵ' denote the threshold values estimated by RDG2 and RDG respectively; $\hat{\lambda}^{max}_{sep}$ denotes the maximum of the floating-point values of the non-linearity term ($\hat{\lambda}$) when the corresponding decision variable subsets are separable.

as $\hat{\lambda}_{int}$) in the decomposition process of RDG2 on f_8 and f_{11}. The value of $\hat{\lambda}_{int}$ consists of the genuine λ value and the computational round-off error. The box plots of the $\hat{\lambda}_{int}$ value from each component are shown in Figure 3a and Figure 3b. We observed that the pattern of the $\hat{\lambda}_{int}$ value was consistent with the pattern of the weight value across the components in CEC'2013 f_8 and f_{11}.

We also recorded the floating-point value of the non-linearity term for separable decision variable subsets ($\hat{\lambda}_{sep}$). It is noteworthy that the $\hat{\lambda}_{sep}$ value represents the computational round-off errors, as the genuine value of λ is zero for separable decision variable subsets. The maximum value of $\hat{\lambda}_{sep}$ from each component is shown in Figure 3c and Figure 3d. In Figure 3c and Figure 3d, we also plotted the threshold values (ϵ) estimated by RDG2, the 25% percentile of the $\hat{\lambda}_{int}$ values and the threshold values (ϵ') estimated by RDG.

On f_{11}, the threshold value estimated by RDG2 was always in between the 25% percentile of the $\hat{\lambda}_{int}$ value and the maximum of $\hat{\lambda}_{sep}$, resulting in 100% decomposition accuracy of RDG2. On the other hand, the RDG method used a global threshold value (ϵ' = 1.52e+06), which was always lower than the maximum of $\hat{\lambda}_{sep}$. That explains why RDG failed to identify the variable interaction structure and placed all the 1000 decision variables into one component.

On f_8, the threshold value estimated by RDG2 was higher than the 25% percentile of the $\hat{\lambda}_{int}$ value for the 11^{th} and 13^{th} components. That is the reason why RDG2 identified the 200 non-separable decision variables in the 11^{th} and 13^{th} components as separable. It is noteworthy that the 25% percentile of the $\hat{\lambda}_{int}$ value from the 11^{th} and 13^{th} components is in the same magnitude of the round-off errors (the maximum of $\hat{\lambda}_{sep}$). Therefore, it is very challenging, if possible, to identify the interacting decision variables in these two components. However, we argue that the 11^{th} and 13^{th} components are not important in terms of the optimization task, as they only contribute marginally to the overall fitness value, as indicated by the weight values.

The number of FEs used by RDG2 was close to the one used by the RDG method on most of the benchmark problems. On f_{11}, the number of FEs used by RDG2 was slightly larger than those by RDG. The reason for this was that RDG failed to identify the variable interaction structure by placing all the decision variables into one group. The DG2 method used a fixed number of FEs ($n^2 + n + 2)/2$ to identify the interaction matrix (all pairwise interactions) of decision variables. Once the interaction matrix is identified, it is possible to generate a more effective decomposition for problems with overlapping components, e.g., f_{13} and f_{14}.

The RDG2, RDG and DG2 methods achieved 100% decomposition accuracy on all of the CEC'2010 benchmark problems. The detailed results were presented in the supplementary material.

4.2.2 Optimization Comparison. Table 2 lists the optimization results of the RDG2, RDG and DG2 methods when embedded into the CC framework to solve the CEC'2013 benchmark problems. The CC-RDG2 algorithm consistently achieved statistically equally well or significantly better solution quality than the other two algorithms.

On most of the benchmark problems where the decomposition results of RDG2 and RDG were similar, CC-RDG2 and CC-RDG performed equally well. However, when the decomposition accuracy of RDG2 was higher than that of RDG, the solution quality generated by CC-RDG2 is potentially better than that generated by CC-RDG, e.g., on f_{11}. Refer to the convergence curve shown in Figure 4c.

The CC-RDG2 algorithm outperformed CC-DG2 on most of the benchmark problems. The main reason for this is that RDG2 used much less FEs than DG2 in the decomposition stage, saving more FEs for the optimization stage. On the benchmark problems where the decomposition accuracy of RDG2 was higher than that of DG2, the CC-RDG2 was able to find statistically significantly better solution quality than the CC-DG2 algorithm, e.g., f_7 and f_8 (Figure 4a and Figure 4b).

It is noteworthy that the DG2 method used more FEs than the RDG method to decompose f_{11}, therefore less FEs were left to actually optimize the problem (the optimization stage). However, the CC-DG2 algorithm generated even better solution quality than CC-RDG (Figure 4c). The reason for this is that DG2 achieved 100% decomposition accuracy on f_{11}, while the decomposition accuracy of RDG was low. This is why DG2 used approximately 50 times more function evaluations than RDG in the decomposition stage. This observation confirms that the decomposition accuracy is of crucial importance in problem decomposition.

The RDG2 method also consistently achieved statistically similar or significantly better solution quality than the other two methods when embedded into the CC framework to solve the CEC'2010 benchmark problems. The detailed optimization results were placed in the supplementary material due to page limits.

5 CONCLUSION

In this paper, we have derived an upper bound on the computational round-off errors involved in calculating the non-linearity term (λ) for the RDG method. This upper bound was then used as the threshold value, and was shown to be able to identify variable interactions across a wide range of benchmark problems. However if the genuine value of the non-linearity term was of the same magnitude as the computational round-off errors, the corresponding variable interaction was difficult to identify. We found that the pattern of the non-linearity term for interacting decision variables was consistent with the pattern of the weight value from each component in two benchmark problems: CEC'2013 f_8 and f_{11}. However, more research needs to be conducted in order to draw any conclusion. Another direction for future work is to generate an effective decomposition for large-scale problems with overlapping components.

Table 2: The optimization results of the RDG2, RDG and DG2 when embedded into the CC framework to solve the CEC'2013 benchmark problems. The entries with the best solution quality are highlighted in bold according to the Wilcoxon rank-sum tests (significance level = 0.05) with Holm p-value correction.

Func	Stats	RDG2	RDG	DG2
f_1	median	**2.76e+05**	**2.84e+05**	5.48e+05
	mean	2.78e+05	2.89e+05	5.51e+05
	std	3.16e+04	3.27e+04	5.87e+04
f_2	median	4.70e+03	4.66e+03	4.69e+03
	mean	4.70e+03	4.68e+03	4.68e+03
	std	2.05e+02	1.77e+02	1.80e+02
f_3	median	**2.04e+01**	**2.03e+01**	2.04e+01
	mean	2.04e+01	2.03e+01	2.04e+01
	std	4.34e-02	4.95e-02	5.21e-02
f_4	median	**5.81e+06**	**5.81e+06**	8.43e+06
	mean	5.83e+06	5.83e+06	8.51e+06
	std	6.32e+05	6.32e+05	8.54e+05
f_5	median	2.24e+06	2.34e+06	2.17e+06
	mean	2.23e+06	2.40e+06	2.18e+06
	std	3.22e+05	4.35e+05	3.51e+05
f_6	median	9.95e+05	9.95e+05	9.95e+05
	mean	9.95e+05	9.96e+05	9.96e+05
	std	6.54e+01	1.47e+02	3.31e+02
f_7	median	**3.12e-19**	**2.93e-20**	1.00e+03
	mean	4.04e-16	8.11e-17	1.05e+03
	std	1.48e-15	2.17e-16	2.78e+02
f_8	median	**8.15e+06**	**8.26e+06**	3.56e+07
	mean	8.70e+06	8.50e+06	3.84e+07
	std	3.61e+06	2.91e+06	1.08e+07
f_9	median	1.74e+08	1.57e+08	1.52e+08
	mean	1.67e+08	1.65e+08	1.51e+08
	std	2.65e+07	4.16e+07	2.86e+07
f_{10}	median	9.05e+07	9.05e+07	9.05e+07
	mean	9.10e+07	9.10e+07	9.13e+07
	std	1.30e+06	1.29e+06	1.50e+06
f_{11}	median	**2.81e+03**	1.68e+07	1.55e+05
	mean	8.68e+03	1.67e+07	2.47e+05
	std	1.24e+04	1.61e+06	2.36e+05
f_{12}	median	1.01e+03	1.01e+03	1.01e+03
	mean	9.81e+02	9.81e+02	1.00e+03
	std	7.30e+01	7.30e+01	5.80e+01
f_{13}	median	**9.04e+05**	2.48e+06	2.27e+06
	mean	9.31e+05	2.46e+06	2.42e+06
	std	1.60e+05	3.82e+05	3.69e+05
f_{14}	median	**2.65e+07**	2.74e+07	3.65e+07
	mean	2.68e+07	2.76e+07	3.58e+07
	std	1.88e+06	1.80e+06	2.85e+06
f_{15}	median	**2.23e+06**	**2.18e+06**	2.93e+06
	mean	2.26e+06	2.19e+06	3.01e+06
	std	2.45e+05	2.28e+05	3.29e+05

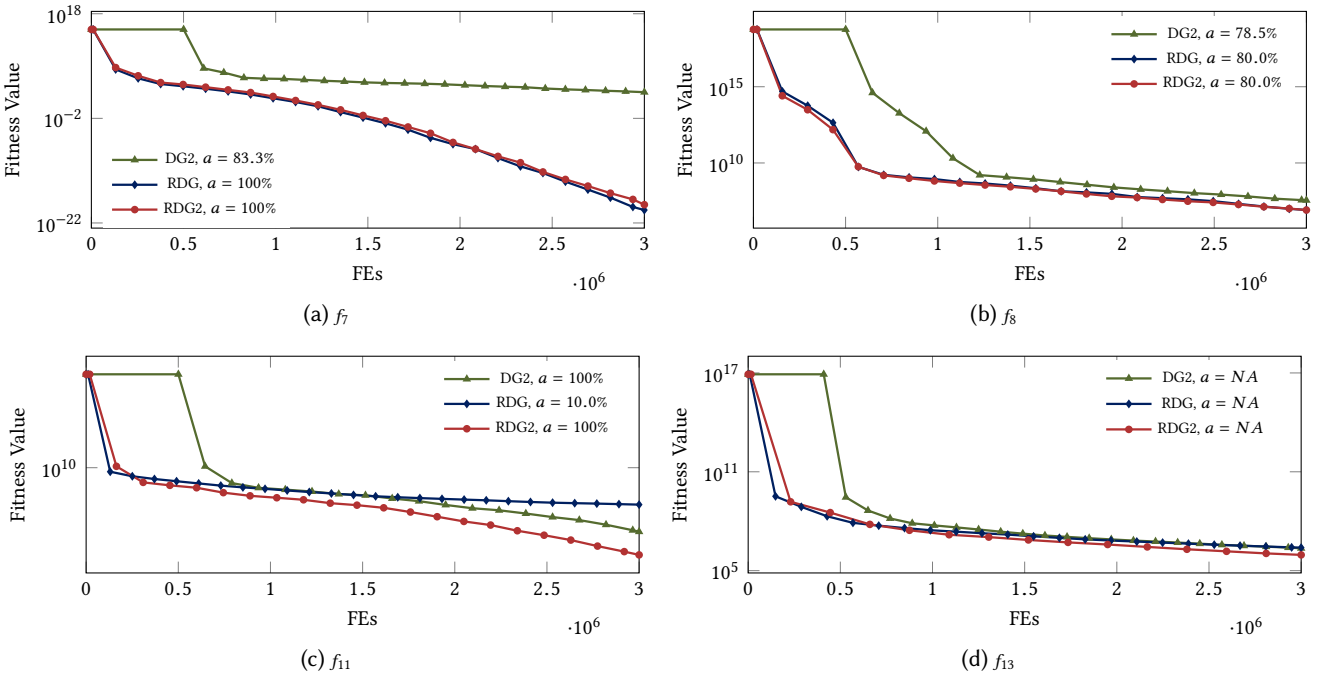

Figure 4: The convergence curves of the RDG2, RDG and DG2 methods when embedded into the CC framework to solve the CEC'2013 f_7, f_8, f_{11} and f_{13} problems. The horizontal axis represents the number of FEs used in the evolutionary process. The vertical axis represents the median of the best fitness found. In the legends, "a" denotes the "accuracy" of decomposition, and "NA" denotes "not applicable".

REFERENCES

[1] 2008. IEEE Standard for Floating-Point Arithmetic. *IEEE Std 754-2008* (Aug 2008), 1–70. https://doi.org/10.1109/IEEESTD.2008.4610935

[2] Wenxiang Chen and Ke Tang. 2013. Impact of problem decomposition on cooperative coevolution. In *2013 IEEE Congress on Evolutionary Computation*. IEEE, 733–740.

[3] Wenxiang Chen, Thomas Weise, Zhenyu Yang, and Ke Tang. 2010. Large-scale global optimization using cooperative coevolution with variable interaction learning. In *Parallel Problem Solving from Nature, PPSN XI*. Springer, 300–309.

[4] Robert M Corless and Nicolas Fillion. 2013. *A graduate introduction to numerical methods*. New York, Springer.

[5] Nikolaus Hansen. 2011. The CMA evolution strategy: A tutorial. *Technique Report* (2011).

[6] Nicholas J Higham. 2002. *Accuracy and stability of numerical algorithms*. SIAM.

[7] Xiao-Min Hu, Fei-Long He, Wei-Neng Chen, and Jun Zhang. 2017. Cooperation coevolution with fast interdependency identification for large scale optimization. *Information Sciences* 381 (2017), 142–160.

[8] Xiaodong Li, Ke Tang, Mohammad N Omidvar, Zhenyu Yang, and Kai Qin. 2013. Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization. *gene* 7, 33 (2013), 8.

[9] Haiyan Liu, Yuping Wang, Xuyan Liu, and Shiwei Guan. 2016. Empirical study of effect of grouping strategies for large scale optimization. In *Neural Networks (IJCNN), 2016 International Joint Conference on*. IEEE, 3433–3439.

[10] Yi Mei, Xiaodong Li, and Xin Yao. 2014. Cooperative coevolution with route distance grouping for large-scale capacitated arc routing problems. *Evolutionary Computation, IEEE Transactions on* 18, 3 (2014), 435–449.

[11] Yi Mei, Mohammad Nabi Omidvar, Xiaodong Li, and Xin Yao. 2016. A competitive divide-and-conquer algorithm for unconstrained large-scale black-box optimization. *ACM Trans. Math. Software* 42, 2 (2016), 13.

[12] Mohammad Nabi Omidvar, Xiaodong Li, Yi Mei, and Xin Yao. 2014. Cooperative co-evolution with differential grouping for large scale optimization. *Evolutionary Computation, IEEE Transactions on* 18, 3 (2014), 378–393.

[13] Mohammad Nabi Omidvar, Xiaodong Li, and Ke Tang. 2015. Designing benchmark problems for large-scale continuous optimization. *Information Sciences* 316 (2015), 419–436.

[14] Mohammad Nabi Omidvar, Ming Yang, Yi Mei, Xiaodong Li, and Xin Yao. 2017. DG2: A faster and more accurate differential grouping for large-scale black-box optimization. *IEEE Transactions on Evolutionary Computation* 21, 6 (2017), 929–942.

[15] Mitchell A Potter and Kenneth A De Jong. 1994. A cooperative coevolutionary approach to function optimization. In *Parallel problem solving from nature PPSN III*. Springer, 249–257.

[16] Eman Sayed, Daryl Essam, Ruhul Sarker, and Saber Elsayed. 2015. Decomposition-based evolutionary algorithm for large scale constrained problems. *Information Sciences* 316 (2015), 457–486.

[17] David J Sheskin. 2003. *Handbook of parametric and nonparametric statistical procedures*. CRC Press.

[18] Pat H Sterbenz. 1974. Floating-point computation. (1974).

[19] Yuan Sun, Michael Kirley, and Saman Kumara Halgamuge. 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*. ACM, 313–320.

[20] Yuan Sun, Michael Kirley, and Saman Kumara Halgamuge. 2017. A recursive decomposition method for large scale optimization. *IEEE Transactions on Evolutionary Computation* (2017). https://doi.org/10.1109/TEVC.2017.2778089

[21] Yuan Sun, Michael Kirley, and Saman K Halgamuge. 2017. Quantifying variable interactions in continuous optimization problems. *IEEE Transactions on Evolutionary Computation* 21, 2 (2017), 249–264.

[22] Ke Tang, X Yao, and Pn Suganthan. 2010. Benchmark functions for the CEC'2010 special session and competition on large scale global optimization. *Technique Report, USTC, Natrue Inspired Computation and Applications Laboratory* 1 (2010), 1–23.

[23] Masaru Tezuka, Masaharu Munetomo, and Kiyoshi Akama. 2004. Linkage identification by nonlinearity check for real-coded genetic algorithms. In *Genetic and Evolutionary Computation–GECCO 2004*. Springer, 222–233.

[24] Frans Van den Bergh and Andries P Engelbrecht. 2004. A cooperative approach to particle swarm optimization. *Evolutionary Computation, IEEE Transactions on* 8, 3 (2004), 225–239.

[25] Zhenyu Yang, Ke Tang, and Xin Yao. 2008. Large scale evolutionary optimization using cooperative coevolution. *Information Sciences* 178, 15 (2008), 2985–2999.