Quantifying Variable Interactions in Continuous Optimization Problems

Yuan Sun, *Student Member, IEEE*, Michael Kirley, *Member, IEEE*, and Saman K. Halgamuge, *Senior Member, IEEE*

***Abstract*—Interactions between decision variables typically make an optimization problem challenging for an evolutionary algorithm (EA) to solve. Exploratory landscape analysis (ELA) techniques can be used to quantify the level of variable interactions in an optimization problem. However, many studies using ELA techniques to investigate interactions have been limited to combinatorial problems, with very few studies focused on continuous variables. In this paper, we propose a novel ELA measure to quantify the level of variable interactions in continuous optimization problems. We evaluated the efficacy of this measure using a suite of benchmark problems, consisting of 24 multidimensional continuous optimization functions with differing levels of variable interactions. Significantly, the results reveal that our measure is robust and can accurately identify variable interactions. We show that the solution quality found by an EA is correlated with the level of variable interaction in a given problem. Finally, we present the results from simulation experiments illustrating that when our measure is embedded into an algorithm design framework, the enhanced algorithm achieves equal or better results on the benchmark functions.**

***Index Terms*—Continuous optimization problem, exploratory landscape analysis (ELA), maximal information coefficient (MIC), variable interaction.**

I. Introduction

In a continuous optimization problem, the objective is to improve a measure of performance or cost—the output variable—by adjusting the values of the input variables, when both input and output variables are real numbers. Such problems are ubiquitous in the real world, for example, engineering control problems; maximizing annual revenue; or building a parameterized model of a physical phenomenon. As the dimensionality of a continuous optimization problem increases, solving the problem can be very challenging, especially when other well-known problem characteristics are considered.

There is abundant evidence suggesting that the interaction between decision variables makes an optimization problem difficult for an evolutionary algorithm (EA) to solve (see [1]–[3] for representative papers on this topic). In a continuous optimization problem, two decision variables interact if they cannot be optimized independently when attempting to find the optimal solution (see Section II-A1 for a formal definition). However, in many practical applications the level of variable interaction is usually unknown, which raises the following research questions.

1) What is the best way to identify the pairwise interaction between decision variables in a continuous optimization problem?
2) How can the level of variable interactions in a continuous optimization problem be measured?
3) Can the level of variable interactions be used to guide the search in continuous optimization problems?

Exploratory landscape analysis (ELA) [4]–[6] is a technique that can be used to explore the fitness landscape [7] of an optimization problem, capturing certain characteristics (e.g., the level of variable interactions) of the problem. Once characteristics are identified, they can be used to predict the problem difficulty [8], [9] or to select an appropriate algorithm to use when solving the optimization problem [10]–[12]. A summary of the well-known ELA measures for continuous optimization problems can be found in [10] and [12]. However, despite many recent efforts, there is a lack of research quantifying the level of interactions between continuous variables and the effects such interactions have on problem difficulty.

There are a number of ELA measures quantifying variable interaction in combinatorial optimization problems. Epistasis variance [13] and epistasis correlation [14] measure the interaction between binary variables by investigating the linear approximation model of the fitness function. However, they have been criticized for only measuring the absence of epistasis [15], [16]. In addition, they do not identify the individual decision variables clearly. Bit-wise epistasis [17] is a more comprehensive technique, investigating the independence of each decision variable. Unfortunately, this technique can only identify independent decision variables, rather than identifying pair-wise interactions between decision variables.

A recently proposed measure, entropic epistasis (EE) [18], identifies interaction between decision variables based on the concept of "mutual information." However, the calculation of

Manuscript received November 24, 2015; revised March 22, 2016 and May 31, 2016; accepted August 7, 2016. Date of publication August 10, 2016; date of current version March 28, 2017. This work was supported by the University of Melbourne. *(Corresponding author: Yuan Sun.)*

Y. Sun is with the Department of Mechanical Engineering, University of Melbourne, Parkville, VIC 3010, Australia (e-mail: yuans2@student.unimelb.edu.au).

M. Kirley is with the Department of Computing and Information Systems, University of Melbourne, Parkville, VIC 3010, Australia (e-mail: mkirley@unimelb.edu.au).

S. K. Halgamuge is with the Research School of Engineering, Australian National University, Canberra, ACT 2601, Australia (e-mail: saman.halgamuge@anu.edu.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2016.2599164

$$\begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \end{array} \begin{array}{c} \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \\ \begin{bmatrix} - & 1 & 0 \\ 1 & - & 0 \\ 0 & 0 & - \end{bmatrix} \end{array}$$

Fig. 1. IM: "1" means "interacting" and "0" means "independent." A "-" on the diagonal entry indicates that it is meaningless to measure the level of interaction between one decision variable and itself.

mutual information is computationally expensive, especially when adapted for continuous optimization problems. The problem of the computational complexity has been solved to some extent via a recently proposed statistical coefficient—the maximal information coefficient (MIC) [19]. Importantly, the MIC can capture functional relationships between two variables.

In this paper, we propose a novel ELA measure, which we call maximum EE (MEE), based on the MIC. The MEE measure can be used to quantify the level of variable interactions in a continuous optimization problem. The MEE measure identifies the interaction matrix (IM) of decision variables in a continuous optimization problem. Each entry in the IM describes the corresponding pairwise interaction between the decision variables, as shown in Fig. 1, where "1" means "interacting" and "0" means "independent." A "-" on the diagonal entry indicates that it is meaningless to measure the level of interaction between one decision variable and itself. MEE identifies interactions between decision variables x_i and x_j by calculating the MIC between x_j and the partial derivative of the objective function with respect to x_i (see Section II-C and/or [19], [20] for the implementation of MIC). The level of variable interactions in the continuous optimization problem is subsequently approximated using three indices derived from the IM.

We evaluate the efficacy of our MEE measure using a suite of 24 benchmark continuous optimization functions with different level of variable interactions. The experimental results show that MEE can identify the IM and accurately quantify the level of variable interactions. We then show that there is a correlation between the MEE level and the solution quality found by an EA. The rationale behind this particular analysis exercise was based on the hypothesis that the level of variable interaction is a factor that makes a continuous optimization problem challenging for some of the EAs to solve [1]–[3].

In the final set of simulation experiments, we investigate whether the MEE measure can be used to effectively guide the search for a (near) optimal solution to a given problem. The algorithm design framework, proposed in [21], employs the Pearson correlation coefficient to quantify the level of variable interactions in an optimization problem. Based on that, appropriate operator(s) is(are) selected to guide the search. We observe that by using MEE, instead of the Pearson correlation coefficient to quantify the level of variable interactions, equal or better results can be achieved when solving the 24 benchmark problems.

The remainder of this paper is organized as follows. Section II summarizes the main problem characteristics, related ELA measures, methods to identify variable interactions, and describes the MIC in detail. Section III describes the proposed MEE measure in detail. Section IV describes experiments to evaluate the proposed MEE measure. Section V presents and analyzes the experimental results. Section VI shows how the MEE measure can be embedded within a framework to design efficient optimization algorithms. Section VII concludes this paper and outlines future directions.

II. Related Work

A. Problem Characteristics and Related ELA Measures

This section describes the main problem characteristics believed to contribute to difficulty of an optimization problem. We also briefly describe the relevant ELA measures that can be used to extract the problem characteristics from an optimization problem.

In a recent survey paper, a detailed analysis of problem characteristics (e.g., separability, modality, basins of attraction, ruggedness, smoothness, and neutrality) related to problem difficulty in continuous optimization problems was presented [12]. Due to space constraints (page limit), we limit our discussion to separability and modality in this paper. In related work, we have attempted to map ELA measures used to capture the problem characteristics [29]. However, other ELA measures such as fitness distance correlation [8] and dispersion metric [30] should be examined to paint a more coherent and complete picture.

1) Separability: In an optimization problem, separability refers to degree of decision variable interaction. It is important to note, that the level of interaction between given decision variables may be different. Take the following objective function as an example:

$$f(\vec{x}) = (x_1 - x_2)^2 + (x_2 - x_3)^2 + x_4^2, \quad \vec{x} \in [-1, 1]^4. \tag{1}$$

Both $\{x_1, x_2\}$ and $\{x_1, x_3\}$ interact with each other. However, x_1 and x_2 interact directly; x_1 and x_3 are linked by x_2. The former is called *direct interaction* and the latter is called *indirect interaction*. The formal definitions of direct interaction and indirect interaction are described as follows [31].

Definition 1: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a differentiable function, where d is the number of decision variables. Decision variables x_i and x_j interact directly if a candidate solution $\vec{x}_*$ exists, such that

$$\frac{\partial^2 f(\vec{x})}{\partial x_i \partial x_j}\Big|_{\vec{x}_*} \neq 0 \tag{2}$$

denoted by $x_i \leftrightarrow x_j$. Decision variables x_i and x_j interact indirectly if for all candidate solutions

$$\frac{\partial^2 f(\vec{x})}{\partial x_i \partial x_j} = 0 \tag{3}$$

and a set of decision variables $\{x_{k1}, \ldots, x_{kt}\} \subset \vec{x}$ exists, such that $x_i \leftrightarrow x_{k1} \leftrightarrow \cdots \leftrightarrow x_{kt} \leftrightarrow x_j$. Decision variables x_i and x_j are independent if for all candidate solutions, (3) holds and does not exist a set of decision variables $\{x_{k1}, \ldots, x_{kt}\} \subset \vec{x}$, such that $x_i \leftrightarrow x_{k1} \leftrightarrow \cdots \leftrightarrow x_{kt} \leftrightarrow x_j$.

Definition 1 describes the direct interaction and indirect interaction based on the partial derivative of a differentiable function. However, if a function is not differentiable, the partial

TABLE I
CLASSIFICATION AND BRIEF DESCRIPTION OF SELECTED METHODS THAT CAN BE USED TO IDENTIFY VARIABLE INTERACTIONS. NOTE THAT THE SUMMARY HERE IS BY NO MEANS EXHAUSTIVE/COMPLETE. WE INTRODUCE TWO METHODS IN EACH CATEGORY TO HELP READERS BETTER UNDERSTAND THE DIFFERENCES BETWEEN EACH CATEGORY

Category	Name	Authors	Brief description
Statistical methods	CBAVP [22]	Ray et al.	Calculates the Pearson correlation coefficient between decision variables based on the 50% high-level candidate solutions in a population.
	EE [18]	Seo et al.	Calculates mutual information between decision variables.
Interaction adaptation	LEGO [23]	Smith et al.	Uses two boolean flags to evolve the variable interactions in the evolutionary process.
	LLGA [24]	Harik et al.	Uses the linkage skew and linkage shift mechanisms to evolve the variable interactions.
Model Building	EDA [25]	Muhlenbein et al.	Builds a probabilistic model based on the promising candidate solutions and updates the model during the evolutionary process. The population for the next generation is sampled from the current model.
	CMA-ES [26]	Hansen et al.	Evolves the mean and covariance matrix of the promising candidate solutions in the evolutionary process. The population for the next generation is sampled from the current mean and covariance matrix.
Perturbation based	DG [27]	Omidvar et al.	Detects the changes in the fitness when adding a perturbation to decision variable x_i. If the changes maintain the same for different values of x_j, then x_i and x_j interact.
	LIMD [28]	Munetomo et al.	Identifies interactions by adding perturbations to a pair of bits simultaneously to detecting the non-monotonicity changes in the fitness.

derivative may not exist. In this case, the partial subderivative [32], [33] can be used as a substitute for the partial derivative, which will be detailed in Section III.

Typical ELA measures quantifying the level of variable interactions in an optimization problem include epistasis variance [13], epistasis correlation [14], bit-wise epistasis [17], EE [18], sign epistasis [34], auto-correlation function [35], and meta-model [5]. In this section, we only describe EE and meta-model in detail, as they will be used for comparison in the numerical experiments.

The EE $\mathcal{E}(V)$ of a nonempty decision variable subset V is defined as follows:

$$\mathcal{E}(V) = \frac{I(X_V, Y) - \sum_{v \in V} I(X_v, Y)}{I(X_V, Y)} \tag{4}$$

where X denotes the decision variable(s), Y denotes the objective function value, v is an individual decision variable in the subset V, and $I(X, Y)$ is the mutual information between X and Y. It is important to note that EE was originally proposed for combinatorial optimization problem. In this paper, the *k-d* partitioning is used to estimate the entropy for multidimensional continuous variables [36]. The computational complexity of EE grows exponentially with the dimensionality d, as the sample size used to estimate d-dimensional entropy should be greater than 2^d [36].

The meta-model method builds a linear or quadratic regression model based on n samples $\{(\vec{x}_i, y_i), 1 \leq i \leq n\}$. The adjusted coefficient of determination ($\bar{R}^2$) is employed as an indicator for model accuracy, which is defined as

$$\bar{R}^2 = 1 - \frac{(n-1)\sum_{i=1}^{n}\left(\hat{y}_i - \bar{y}\right)^2}{(n-p-1)\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5}$$

where $\hat{y}_i$ is the corresponding estimation made by the regression model: $\hat{y}_i = f(\vec{x}_i)$; $\bar{y}$ is the mean of y_i; p is the degree of the polynomial regression model (linear: $p = 1$, quadratic: $p = 2$). Green [37] suggested that the sample size n should be greater than $50 + 8d$, where d is the number of variables in $\vec{x}_i$.

2) Modality: Modality refers to global or local optima in an optimization problem. Without loss of generality, we only consider minimization problems in this paper. The definitions of global and local optima are described as follows [12].

Definition 2: In an objective function $f : \mathbb{R}^d \to \mathbb{R}$, a candidate solution $\vec{x}_o$ is a global optimum if for all $\vec{x} \in \mathbb{R}^d$, $f(\vec{x}_o) \leq f(\vec{x})$. A candidate solution $\vec{x}_l$ is a local optimum if there exists a $\delta > 0$ such that for all $x \in B(\vec{x}_l, \delta), f(\vec{x}_l) \leq f(\vec{x})$, where $B(\vec{x}_l, \delta)$ is the ball of center $\vec{x}_l$ and radius δ (excluding $\vec{x}_l$); If $f(\vec{x}_l) < f(\vec{x})$, for all $x \in B(\vec{x}_l, \delta)$, $\vec{x}_l$ becomes a strict local optimum.

An optimization problem is unimodal if it only has one optimum, and is multimodal if it has more than one optimum. Several landscape features related to modality may influence the search difficulty, e.g., the number of local optima, the size of the optimum attraction basin, the average shortest path between local optima, ruggedness, smoothness, and neutrality [12], [38]–[41]. Representative examples of ELA measures used to quantify modality include the information content of fitness sequences [11], local optima network [38], expected number of local optima [42], and the funnel structure detection [43].

B. Methods Identifying Variable Interaction

A large number of methods have been proposed to identify variable interactions in different domains (e.g., large scale global optimization, evolutionary computation research). In this section, we classify these methods into four categories and discuss each category in detail.

Yu *et al.* [44] classified the methods identifying variable interactions into three categories: 1) perturbation; 2) interaction adaptation; and 3) model building. Omidvar *et al.* [27] added a fourth category, based in "random grouping" [45], referred to as a random method. We argue that this method should not be regarded as a variable interaction identifying method as it assigns decision variables into subgroups randomly. Consequently, we introduce a new category based on statistical measures, which we call "statistical methods." A summary and brief description of the selected methods identifying variable interactions are presented in Table I. Note that the review in Table I is by no means complete. We select two

methods in each category in order to help the readers better understand the differences between each category.

Statistical methods identify variable interactions by calculating statistical measures based on the "promising" candidate solutions. For example, CBAVP [22] calculates the Pearson correlation coefficient between two decision variables based on the promising candidate solutions of a population. If the coefficient is greater than a given threshold, the two decision variables are classified as interacting with each other. However, the Pearson correlation coefficient can only capture linear relationship between two decision variables. EE [18] identifies variable interactions by investigating the mutual information between decision variables. The MEE measure proposed in this paper is based on the statistical coefficient MIC, therefore it falls into the newly proposed category.

Perturbation methods identify variable interactions by adding a small perturbation to the decision variable and detecting the changes in the objective function. Representative examples include differential grouping (DG) [27] and linkage identification by nonlinearity detection [28]. These methods are sensitive to the computational errors and noises in the system. Smooth landscapes pose additional challenges for these methods; the addition of a perturbation to the decision variable may not result in significant changes in the objective functions and thus may be difficult to detect.

Model building methods generate a probabilistic model as part of the evolutionary process. For example, the class of estimation of distribution algorithms (EDAs) [25] builds a probabilistic model based on the promising candidate solutions and updates the model during each generation. The candidate solutions for the next generation are sampled from the current probabilistic model. The covariance matrix adaptation—evolutionary strategy (CMA-ES) [26] evolves the mean and covariance matrix of the promising candidate solutions. The population for the next generation is subsequently generated from the mean and covariance matrix.

Interaction adaptation methods identify variable interactions in the evolutionary process. For example, The linkage learning genetic algorithm (LLGA) [24] uses the linkage skew and linkage shift mechanisms to evolve the IM of the decision variables. The aim of the two mechanisms is to place the interacting decision variables closer in the chromosome. The chromosome with closer interacting decision variables has a higher survival rate under recombination.

C. Maximal Information Coefficient

Typical measures of correlation between two variables, such as the Pearson correlation coefficient, capture only linear relationships. In contrast, it is possible to measure statistical dependence between two variables, without assuming the relationship is linear, using the information-theoretic mutual information [46]. The mutual information between two continuous random variables X and Y is defined as

$$\mathrm{MI}(X, Y) = \int_{x\in X}\int_{y\in Y} p(x, y)\log\left(\frac{p(x, y)}{p(x)p(y)}\right) \tag{6}$$

where $p(x)$ and $p(y)$ are the marginal PDFs of variables X and Y, and $p(x, y)$ is the joint PDF of X and Y. Mutual information has been used, for example, to detect phase transitions in a variety of systems, including socio-economic systems [47].

Estimating mutual information accurately from limited data is a difficult problem, especially for continuous variables [48] (see [49] for a review of methods). Consequently, the MIC, was recently introduced to address this problem [19]. MIC is part of the maximal information-based nonparametric exploration suite, which uses "bins" as a means to apply mutual information on continuous random variables, thus capturing a broad range of associations both functional and not. In this paper, a function relationship means a distribution (X, Y) in which Y is a function (or a superposition of several functions) of X, potentially with independent noise added. In the continuous optimization domain, most of the optimization problems investigated are functions. Therefore, only functional relationships are considered in this paper.

MIC computes the mutual information between two variables at a variety of scales and finds the largest possible mutual information at any scale. Let D denote a set of ordered pairs, $\{(x_i, y_i), i = 1, \ldots, n\}$, G denote a $m \times n$ grid covering D. That means dimensions x and y are partitioned into m and n intervals, respectively. The PDF of a grid cell is proportional to the number of data points inside that cell. The characteristic matrix $M(D)_{m,n}$ represents the highest normalized mutual information of D with the $m \times n$ partition, which is defined as

$$M(D)_{m,n} = \frac{\max(\mathrm{MI})}{\log\min(m, n)} \tag{7}$$

where max(MI) is the maximum mutual information of D by all possible $m \times n$ partitions. The MIC of a set D is then defined as

$$\mathrm{MIC}(D) = \max_{0<mn<B(N)}\left\{M(D)_{m,n}\right\} \tag{8}$$

where N is the sample size, and the function $B(N) = N^{0.6}$ was heuristically determined by Reshef *et al.* [19]. It represents the maximal value in the characteristic matrix $M(D)$ subject to $0 < mn < B(N)$.

It should be noted that the MIC tends to 1 for functional relationships with probability approaching 1 as the sample size grows, and converges to 0 for statistically independent variables [19].

III. Measuring Variable Interaction Using Maximal Information Coefficient

In this section, we propose a novel ELA measure, which we call MEE, to quantify the level of variable interactions in a continuous optimization problem.

Lemma 1: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a differentiable function. If $(\partial f/\partial x_i)$ and x_j have a functional relationship, x_i and x_j interact directly. Otherwise, x_i and x_j interact indirectly or are independent.

Proof: Without loss of generality, we assume that

$$\frac{\partial f}{\partial x_i} = g(\vec{x}_t) \tag{9}$$

where $\vec{x}_t \subset \vec{x}$, $g(\vec{x}_t)$ is a function of $\vec{x}_t$. If $(\partial f/\partial x_i)$ is a function of x_j, then $x_j \in \vec{x}_t$. By taking the derivative with respect to x_j, we can obtain

$$\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial g(\vec{x}_t)}{\partial x_j} \neq 0. \tag{10}$$

According to Definition 1, x_i and x_j interact directly. On the other hand, if $(\partial f/\partial x_i)$ is not a function of x_j, then $x_j \notin \vec{x}_t$. By taking the derivative with respect to x_j, we can obtain

$$\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial g(\vec{x}_t)}{\partial x_j} = 0. \tag{11}$$

According to Definition 1, x_i and x_j interact indirectly or are independent. ■

Lemma 2: Two variables X and Y have a functional relationship, if $\lim_{n\to\infty} \mathrm{MIC}(X, Y) = 1$; two variables X and Y are independent, if $\lim_{n\to\infty} \mathrm{MIC}(X, Y) = 0$, where n is the sample size, $\mathrm{MIC}(X, Y)$ is the MIC of X and Y [19].

Proposition 1: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a differentiable function. If $\lim_{n\to\infty} \mathrm{MIC}((\partial f/\partial x_i), x_j) = 1$, x_i and x_j interact directly; if $\lim_{n\to\infty} \mathrm{MIC}((\partial f/\partial x_i), x_j) = 0$, x_i and x_j interact indirectly or are independent.

Proof: Refer to Lemmas 1 and 2. ■

Proposition 1 shows a straightforward way to identify the direct interaction between decision variable pairs in a differentiable function. If the MIC between $\partial f/\partial x_i$ and x_j is greater than a given threshold, then x_i and x_j can be regarded as directly interacting with each other. However, in a nondifferentiable function, the partial derivative $(\partial f/\partial x_i)$ may not exist. Therefore, Proposition 1 cannot be applied directly to identify variable interactions in a nondifferentiable function. In this case, the partial subderivative $(\hat{\partial} f/\partial x_i)$ [32] can be used as a replacement for the partial derivative $(\partial f/\partial x_i)$.

The partial subderivative $(\hat{\partial} f/\partial x_i)$ is defined as follows:

$$\frac{\hat{\partial} f}{\partial x_i} = \left\{ v \,\middle|\, \liminf_{\delta x_i \to 0} \frac{f(x_i + \delta x_i) - f(x_i) - v \cdot \delta x_i}{|\delta x_i|} \geq 0 \right\}. \tag{12}$$

The lim inf denotes the *lower limits*, defined as

$$\liminf_{x \to \bar{x}} \varphi(x) = \lim_{\delta \to 0} \left(\inf_{x \in B(\bar{x}, \delta)} \varphi(x) \right) \tag{13}$$

where $B(\bar{x}, \delta)$ is a ball of center $\bar{x}$ and radius δ (excluding $\bar{x}$), and φ is a function. Each $v \in \hat{\partial} f/\partial x_i$ is a first-order partial subderivative of f with respect to x_i.

In this paper, an optimization problem is regarded as a "black-box" where partial (sub)derivative information is not known. Therefore, we approximate this value using the expression

$$\frac{\partial f}{\partial x_i} \text{ or } \frac{\hat{\partial} f}{\partial x_i} \approx \frac{f(x_i + \delta x_i) - f(x_i)}{\delta x_i} \tag{14}$$

where δx_i is a small number. In this paper, we use a default value of $\delta x_i = 10^{-6}$. It is important to note that the estimation of partial (sub)derivatives consumes additional function evaluations, which will be analyzed in the end of this section.

The direct interaction between decision variables can be identified by Proposition 1. The question then, is: "How to identify indirect interaction between decision variables?"

Algorithm 1 MEE: Identifying Direct Interaction

Require: f, d, $\vec{ub}$, $\vec{lb}$, n, α, β, δx
1: **for** $i, j = 1$ **to** d **and** $i \neq j$ **do**
2: $IM(i, j) \leftarrow 0$ // Initialize the interaction matrix IM
3: **end for**
4: **for** $i = 1$ **to** d **do**
5: **for** $j = i + 1$ **to** d **do**
6: $\vec{x}_0 \leftarrow (\vec{ub} - \vec{lb})rand(d, 1) + \vec{lb}$
7: $\vec{xj} \leftarrow \left(\vec{ub}(j) - \vec{lb}(j)\right)rand(1, n) + \vec{lb}(j)$ // Randomly generate n values from dimension x_j.
8: **for** $k = 1$ **to** n **do**
9: $\vec{x} \leftarrow \vec{x}_0$
10: $\vec{x}(j) \leftarrow \vec{xj}(k)$
11: $y_1 \leftarrow f(\vec{x})$
12: $\vec{x}(i) \leftarrow \vec{x}(i) + \delta x$
13: $y_2 \leftarrow f(\vec{x})$
14: $\vec{de}(k) \leftarrow \frac{y_2 - y_1}{\delta x}$
15: **end for**
16: $ave\_de \leftarrow mean(\vec{de})$
17: **for** $k = 1$ **to** n **do**
18: **if** $|(\vec{de}(i) - ave\_de)| < \beta$ **then**
19: $\vec{de}(i) \leftarrow ave\_de$
20: **end if**
21: **end for**
22: **if** $MIC(\vec{de}, \vec{x}_j) > \alpha$ **then**
23: $IM(i, j) \leftarrow 1$, $IM(j, i) \leftarrow 1$
24: **end if**
25: **end for**
26: **end for**
27: **return** IM as IM_d // The interaction matrix only with direct interaction.

According to Definition 1, if there is not a direct interaction between x_i and x_j, but we can find a set of interacting decision variables to link them together, then x_i and x_j are regarded as indirectly interacting with each other. Inspired by this, we propose a method to comprehensively capture both direct and indirect interactions between decision variables.

The proposed algorithm consists of two stages. The first stage is to identify direct interaction between decision variables (Algorithm 1), and the second stage is to identify indirect interaction between decision variables.

In Algorithm 1, the inputs are: f is the objective function, d is the dimensionality, $\vec{ub}$ and $\vec{lb}$ are the upper and lower bounds of the search space, n is the sample size, α is the threshold to identify direct interaction between pairwise decision variables, β is employed to remove the computational errors in the system, and δx is used to calculate the partial derivatives.

Algorithm 1 begins by initializing the IM to 0. The interaction matrix is a $d \times d$ symmetric matrix, which contains the information of all pairwise interactions between decision variables. $\mathrm{IM}(i, j) = 1$ means decision variables x_i and x_j interact with each other; while $\mathrm{IM}(i, j) = 0$ means x_i and x_j are independent.

For each pair of decision variables (x_i, x_j), $1 \leq i < j \leq d$, the direct interaction is identified as follows. It begins by randomly generating a candidate solution $\vec{x}_0$ and an array of n elements $\vec{xj}$ from the search space (lines 6 and 7). First, the algorithm substitutes the jth element of $\vec{x}_0$ with the first element of $\vec{xj}$ to make a new candidate solution $\vec{x}$ (line 10). Then the algorithm calculates the partial derivative of f with respect

to x_i and puts it in the first element of $\vec{de}$. Second, the algorithm substitutes the jth element of $\vec{x}_0$ with the second element of $\vec{xj}$ and obtains a new candidate solution $\vec{x}$. Again the algorithm calculates $\partial f/\partial x_i$ and puts it in the second element of $\vec{de}$. This process continues until all elements in $\vec{xj}$ are used and n elements of $\vec{de}$ are obtained. $\partial f/\partial x_i$ is calculated as follows: calculate the fitness value of $\vec{x}$, denoted by y_1; add a small value δx to the ith element of $\vec{x}$ and calculate the fitness value again, denoted by y_2; calculate $(y_2 - y_1)/\delta x$ as the approximation of $\partial f/\partial x_i$ (lines 11–14).

If x_i and x_j do not directly interact with each other, theoretically $\partial f/\partial x_i$ should maintain the same value when x_j changes. However, due to numerical computational errors, $\partial f/\partial x_i$ may fluctuate slightly around the true value. The small computational errors of the partial derivative in turn may result in large computational errors of the MIC between $\partial f/\partial x_i$ and x_j. This may significantly affect the accuracy of identifying interacting variables. In order to remove the computational errors, the parameter β is employed, such that if the difference between $\partial f/\partial x_i$ and the mean of the n number of $\partial f/\partial x_i$ is less than β, the algorithm sets $\partial f/\partial x_i$ to the mean (lines 16–21).

Then, the algorithm calculates the MIC between the filtered $\vec{de}$ and the $\vec{xj}$. If MIC($\vec{de}, \vec{xj}$) is greater than or equal to a given threshold α, then x_i and x_j are regarded as directly interacting with each other, and the corresponding entries of the IM(i, j) as well as IM(i, j) are set to 1 (lines 22–24).

Once the IM$_d$ (IM only with direct interactions) is identified, the second stage is deployed, attempting to identify the indirect interactions. This stage begins by constructing an interaction graph based on the IM$_d$: setting each decision variable x_i as a vertex i; connecting vertices i and j if IM$_d(i, j) = 1$. Then the breadth first search algorithm is employed to search for the strongly connected components (SCCs) in the graph. Finally, all the pairs of vertices i and j in each identified SCC are connected and the corresponding IM(i, j) and IM(j, i) are set to 1. The algorithm returns the IM as the output.

With the IM and IM$_d$ identified, three measures can be calculated: 1) the degree of direct variable interaction (DDVI); 2) the degree of indirect variable interaction (DIVI); and 3) the degree of variable interaction (DVI)

$$\text{DDVI} = \frac{\sum_{1\le i\ne j\le d} \text{IM}_d(i,j)}{d(d-1)}$$
$$\text{DVI} = \frac{\sum_{1\le i\ne j\le d} \text{IM}(i,j)}{d(d-1)}$$
$$\text{DIVI} = \text{DVI}-\text{DDVI}.$$

We conclude this section with a brief discussion of the associated computational cost (complexity) of our approach. For each pair of decision variables, the computational cost when examining direct interactions is $2n$ with respect to function evaluations, where n is the sample size. In a d-dimensional problem, the total number of decision variable pairs is $d(d-1)/2$. Therefore, the total number of function evaluations used to obtain the IM is $nd(d-1)$. Note that the second stage of MEE does not use any function evaluation.

TABLE II
EXTENDED BENCHMARK SUITE WITH SIX CATEGORIES AND 24 BENCHMARK FUNCTIONS. THE BASE FUNCTIONS ARE PRESENTED IN TABLE III. R REPRESENTS ROTATION. THE GLOBAL MINIMA FOR ALL BENCHMARK FUNCTIONS ARE $f(\vec{0}) = 0$

Type	Func	Equation
I	f_1	$f(x) = sphere(x)$
	f_2	$f(x) = elli(x)$
	f_3	$f(x) = rast(x)$
	f_4	$f(x) = alpi(x)$
II	f_5	$f(x) = sphe(x(1:d/2)) + ackl(Rx(d/2+1:d))$
	f_6	$f(x) = elli(x(1:d/2)) + schw(x(d/2+1:d))$
	f_7	$f(x) = rast(x(1:d/2)) + rast(Rx(d/2+1:d))$
	f_8	$f(x) = alpi(x(1:d/2)) + whit(Rx(d/2+1:d))$
III	f_9	$f(x) = \sum_{i=1}^{d/2} scha(x(2i-1:2i))$
	f_{10}	$f(x) = \sum_{i=1}^{d/2} levi(x(2i-1:2i))$
	f_{11}	$f(x) = \sum_{i=1}^{d/2} thr\_hum(x(2i-1:2i))$
	f_{12}	$f(x) = \sum_{i=1}^{d/2} rose(x(2i-1:2i))$
IV	f_{13}	$f(x) = whit(x(1:d/2)) + whit(x(d/2+1:d))$
	f_{14}	$f(x) = grie(x(1:d/2)) + grie(Rx(x/2+1:d))$
	f_{15}	$f(x) = rast(Rx(1:d/2)) + schw(Rx(d/2+1:d))$
	f_{16}	$f(x) = elli(Rx(1:d/2)) + alpi(Rx(d/2+1:d))$
V	f_{17}	$f(x) = rose(x)$
	f_{18}	$f(x) = \sum_{i=1}^{d-1} grie(x(i:i+1))$
	f_{19}	$f(x) = \sum_{i=1}^{d/2+1} ackl(Rx(i:i+d/2-1))$
	f_{20}	$f(x) = rose(x(1:d/2)) + rose(x(d/2+1:d))$
VI	f_{21}	$f(x) = rast(Rx)$
	f_{22}	$f(x) = whit(Rx)$
	f_{23}	$f(x) = schw(x) + schw(Rx)$
	f_{24}	$f(x) = rast(Rx) + rose(Rx)$

IV. EXPERIMENTAL METHODOLOGY

In this section, detailed numerical experiments are conducted to evaluate the efficacy of the proposed MEE measure. Three research questions guide the experimental design.

Q1. Is it possible to accurately measure the IM of a continuous optimization problem using MEE? (Section IV-C)
Q2. Can the MEE measure be used to quantify the DVI in a continuous optimization problem? (Section IV-D)
Q3. Is the MEE measure correlated with the solution quality found by an EA? (Section IV-E)

A. Benchmark Functions

To adequately investigate the three research questions, we extend the benchmark functions from CEC'2013 special session on large scale global optimization [50]. The original benchmark suite consisted of 15 benchmark functions with fixed dimensionality (1000 or 905). We extend the suite to 24 benchmark functions with tunable dimensionality, which is more reliable and flexible to evaluate the methods identifying variable interactions. Operators such as rotating, shifting, symmetry breaking, and adding ill-condition [3] are also available in this extended benchmark suite.

The extended benchmark suite consists of six categories with 24 functions in total.

1) Fully separable functions: f_1–f_4.

TABLE III
BASE FUNCTIONS: THE DIMENSIONALITY OF THE FIRST NINE BASES FUNCTIONS (FROM SPHERE TO WHITELY) ARE TUNABLE, WHILE THE LAST THREE FUNCTIONS (SCHAFFER, LEVI, AND THREEHUMP) ARE 2-D

Name	Equation
Sphere	$sphe(x) = \sum_{i=1}^{d} x_i^2$
Elliptic	$elli(x) = \sum_{i=1}^{d} 10^{3\frac{i-1}{d-1}} x_i^2$
Rastrigin	$rast(x) = 10d - 10\sum_{i=1}^{d} \cos(2\pi x_i) + \sum_{i=1}^{d} x_i^2$
Alpine	$alpi(x) = \sum_{i=1}^{d} \lvert x_i \sin x_i + 0.1x_i \rvert$
Ackley	$ackl(x) = 20 - 20\exp\left(-0.2\sqrt{\frac{\sum_{i=1}^{d} x_i^2}{d}}\right) - \exp\left(\frac{\sum_{i=1}^{d} 2\pi x_i}{d}\right) + e$
Schwefel_1.2	$schw(x) = \sum_{i=1}^{d} \left(\sum_{j=1}^{i} x_j\right)^2$
Rosenbrock	$rose(x) = \sum_{i=1}^{d-1} 100(x_i^2 - x_{i+1})^2 + \sum_{i=1}^{d-1} (x_i - 1)^2$
Griewank	$grie(x) = 1 + \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos\frac{x_i}{\sqrt{i}}$
Whitley	$whit(x) = \sum_{i=1}^{d}\sum_{j=1}^{d} \frac{1}{4000}(100(x_i^2 - x_j)^2 + (1 - x_j)^2)^2 - \cos(100(x_i^2 - x_j)^2 + (1 - x_j)^2) + 1$
Schaffer	$scha(x) = 0.5 + \frac{\sin(x_1^2 - x_2^2)^2 - 0.5}{1 + 0.001(x_1^2 + x_2^2)^2}$
Levi	$levi(x) = \sin(3\pi x_1)^2 + (x_1 - 1)^2(1 + \sin(3\pi x_2)^2) + (x_2 - 1)^2(1 + \sin(2\pi x_2))^2$
ThreeHump	$thr\_hum(x) = 2x_1^2 - 1.05x_1^4 + \frac{x_1^6}{6} + x_1x_2 + x_2^2$

2) Partially separable functions with $d/2$ separable subcomponents: f_5–f_8.
3) Partially separable functions with $d/2$ nonseparable subcomponents: f_9–f_{13}.
4) Partially separable functions with 2 nonseparable subcomponents: f_{14}–f_{16}.
5) Overlapping functions: f_{17}–f_{20}.
6) Fully nonseparable functions: f_{21}–f_{24}.

The details of the 24 benchmark functions and base functions are presented in Tables II and III. The letter *R* represents the rotation operator, which is employed to generate interactions between decision variables. The condition number for all of the benchmark functions is 100 except for f_1. The optima of all the benchmark functions are shifted to $f(\vec{0}) = 0$.

As the interactions between decision variables are known for the benchmark functions, we can use this suite of functions to investigate the accuracy of any method used to identifying variable interactions.

B. Performance Metrics

Standard machine learning performance metrics are introduced to evaluate the performance of our proposed method used to identify variable interactions: *precision*, *recall*, and *F-score* (Q1). *Precision* measures the accuracy of identifying interacting decision variables, while *recall* measures the comprehensiveness of identifying interacting decision variables. The *F-score* is the harmonic mean of *precision* and *recall*.

Definition 3: For a given objective function $f(\vec{x})$, $\vec{x} \in \mathbb{X}$, and a method ϕ identifying variable interactions, let IM_f be the true IM of $f(\vec{x})$, and IM_ϕ be the IM returned by ϕ. Then the *precision*, *recall*, and *F-score* of ϕ on f are defined as follows:

$$\text{precision} = \frac{\sum_{1\le i\ne j\le d} \text{IM}_f(i,j) \cdot \text{IM}_\phi(i,j)}{\sum_{1\le i\ne j\le d} \text{IM}_\phi(i,j)} \tag{15}$$

$$\text{recall} = \frac{\sum_{1\le i\ne j\le d} \text{IM}_f(i,j) \cdot \text{IM}_\phi(i,j)}{\sum_{1\le i\ne j\le d} \text{IM}_f(i,j)} \tag{16}$$

$$F\text{-score} = \frac{\sum_{1\le i\ne j\le d} 2 \cdot \text{IM}_f(i,j) \cdot \text{IM}_\phi(i,j)}{\sum_{1\le i\ne j\le d} \text{IM}_\phi(i,j) + \text{IM}_f(i,j)}. \tag{17}$$

Example 1: For the objective function listed in (1), the true IM (IM_f) is shown in the left-hand side of the following equation:

$$\begin{bmatrix} - & \mathbf{1} & 1 & 0 \\ \mathbf{1} & - & 1 & 0 \\ 1 & 1 & - & 0 \\ 0 & 0 & 0 & - \end{bmatrix} \begin{bmatrix} - & \mathbf{1} & 0 & 1 \\ \mathbf{1} & - & 0 & 0 \\ 0 & 0 & - & 0 \\ 1 & 0 & 0 & - \end{bmatrix}. \tag{18}$$

Assume the IM returned by ϕ (IM_ϕ) is shown on the right hand side of (18). Entries equal to 1 in both IM_f and IM_ϕ are highlighted in bold. Then, *precision* $= 1/2$; *recall* $= 1/3$; and *F-score* $= 2/5$.

C. Identification of Pairwise Interaction (Q1)

In this section, we describe numerical experiments used to evaluate the efficacy of the MEE method in terms of identifying IM.

The MEE is calculated for each of the benchmark functions proposed in Section IV-A, and evaluated by the performance metrics introduced in Section IV-B. The dimensionality of the benchmark functions are set to $d = 10$. The parameter setting

TABLE IV
PARAMETER SETTINGS. THE VALUES EMPHASIZED IN BOLD ARE THE DEFAULT PARAMETER SETTINGS

Parameter	Selected values
α	$0.1, \mathbf{0.2}, 0.3, 0.4, 0.5, 0.6$
β	$10^{-1}, 10^{-2}, \mathbf{10^{-3}}, 10^{-4}, 10^{-5}, 10^{-6}$
n	$10, \mathbf{50}, 100, 200, 300, 400$
δx	$10^{-3}, 10^{-4}, 10^{-5}, \mathbf{10^{-6}}, 10^{-7}, 10^{-8}$

for MEE is $n = 50$, $\alpha = 0.2$, $\beta = 10^{-3}$, and $\delta x = 10^{-6}$. For each benchmark function and each method, 25 independent runs are conducted. The average *precision*, *recall*, and *F-score* values for the 25 independent runs are recorded. The performance of MEE is compared with DG [27], a state-of-the-art method used to identify variable interactions. The parameter setting for DG is consistent with the original paper.

As part of the analysis, we replace the MIC values with the Pearson correlation coefficient and the Spearman's rank correlation coefficient when examining the pairwise interactions between decision variables [in Algorithm 1 (line 22), simply change MIC$(\vec{de}, \vec{x}_j) > \alpha$ to *PeaCorr*$(\vec{de}, \vec{x}_j) > \alpha$ or *SpeRanCorr*$(\vec{de}, \vec{x}_j) > \alpha$]. We refer to the model using Pearson correlation coefficient/Spearman's rank correlation coefficient as "PCC"/"SRC" in the results section. MEE is compared with PCC and SRC on the 24 benchmark functions. The parameter setting for PCC and SRC are exactly the same as those used in the MEE experiments.

The Kruskal–Wallis nonparametric one-way ANOVA test [51] with 95% confidence interval is employed to determine whether at least one method is significantly different from the others. Then a series of Wilcoxon rank-sum tests ($\alpha = 0.05$) with Holm p-value correction [51] are conducted in a pairwise fashion to find the best performing method.

To test the sensitivity of MEE to the parameters, six different values for each parameter are selected, as shown in Table IV. The values emphasized in bold are the default parameter setting. To test the sensitivity of MEE to each parameter, we set other parameters to the default values and set this parameter to the six different values one by one. For example, to test the sensitivity of MEE to α, we set $\beta = 10^{-3}$, $n = 50$, $\delta x = 10^{-6}$, and let $\alpha = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$ one by one. Thus, $6 \times 4 = 24$ parameter settings were tested. To test the pairwise interaction between parameters, we vary the parameter pair over all possible combinations of values ($6 \times 6 = 36$) and set other parameters to the default values. For each of the parameter settings, MEE is calculated for each of the 24 benchmark functions with 25 independent runs. Therefore, $24 \times 25 = 600$ independent runs are conducted for each parameter setting. The average values for *precision*, *recall*, and *F-score* of the 600 independent runs are recorded and used when evaluating the parameter sensitivity of MEE.

D. Quantification of Variable Interaction (Q2)

In this section, we describe numerical experiments used to evaluate the efficacy of the MEE measure in terms of quantifying the level of variable interactions.

The MEE method is used to calculate the three measures: 1) DDVI; 2) DIVI; and 3) DVI for each of the 24 benchmark functions using the default parameter setting (highlighted in bold in Table IV). The MEE method is compared with EE and Meta-model methods (see Section II-A1 for definitions). The EE method uses mutual information to quantify the level of variable interactions. The *k-d* partitioning is used to estimate the mutual information between continuous variables [36]. The sample size used to estimate d-dimensional entropy should be greater than 2^d [36]. As the 24 benchmark functions are all 10-D, we set the sample size $n = 5000$ for EE. The Meta-model method builds a linear or quadratic regression model based on n samples. It has been suggested that the sample size n should be greater than $50 + 8d$ [37]. To achieve reliable results, we set the sample size $n = 100d$. Both linear and quadratic regression models with interaction effects are considered in this paper, and the corresponding adjusted coefficient of determination ($\bar{R}_l^2$ and $\bar{R}_q^2$) are calculated. For each function and each method, 25 independent runs are conducted to remove randomness, and the mean value of each measure is recorded. For each measure, the Kruskal–Wallis nonparametric one-way ANOVA test [51] with 95% confidence interval is employed to determine whether consistent results can be obtained across the functions in each category.

E. Correlation With Algorithm Performance (Q3)

It is well-known that the interaction between decision variables is a factor that makes an optimization problem difficult for some of the EAs to solve [1]–[3]. Thus it is reasonable to assume that as the level of variable interactions increases in an optimization problem, it may become more difficult for an EA to solve. If a measure successfully quantifies the level of variable interactions, therefore it should be correlated with the algorithm performance. In this section, we describe numerical experiments used to investigate the correlation between MEE and algorithm performance.

Three fully separable benchmark functions—Elliptic, Rastrigin, and Alpine—are used in this section (see Table III). Separable functions are selected, as it is possible to set the level of variable interactions by rotating a subset of the decision variables [3]. To the best of our knowledge, there is no technique available to control the level of variable interactions in a fully nonseparable function. For each benchmark function, we set $d = 100$ and vary the number of interacting decision variables over $\{5i,\ 0 \leq i \leq 20\}$. Therefore, for each benchmark function, 21 instances are obtained with different level of variable interactions (number of interacting decision variables). The optimal solution for all the instances is $f(\vec{0}) = 0$. Note that for each benchmark function, the scale of the 21 instances are the same as rotation does not change the scale of a function.

There are generally two ways to measure algorithm performances. The first way is to use the expected running time, which is the expected number of function evaluations used to find a target solution at the first time. An alternative way is to compare the quality of the best solution found within a given computational budget. In the situation that it is difficult to find

TABLE V
PARAMETER SETTINGS FOR THE SELECTED OPTIMIZERS

Optimizer	Parameter settings
DE	Population size = 50, mutation factor = 0.1, crossover rate = 0.5, mutation strategy = DE/rand/1, crossover strategy = binomial, maximal iteration = 2000.
GA	Population size = 50, crossover rate = 0.5, mutation rate = 0.1, mutation strategy = adding a small number, crossover strategy = uniform, maximal iteration = 2000.
PSO	Population size = 50, velocity clamping factor = 2, individual learning rate = 2, social parameter = 2, minimal inertial weight = 0.4, maximal inertial weight = 0.9, maximal iteration = 2000.
CMA-ES	Population size = 4 + floor(3*log(d)), parent number = floor(population size/2), maximal function evaluation = 10^5.

a target solution, the second criterion is often used to measure algorithm performance. The benchmark functions used in this section are high-dimensional (100) and difficult for optimizers to solve, therefore, we use the best solution found within the fixed number of function evaluations as the measurement of algorithm performance.

MEE is used to calculate the DVI for each instance with default parameter setting. We treat direct interaction and indirect interaction as equally important to problem difficulty. Therefore, only the DVI measure is considered in this section. Four widely used EAs: 1) *CMA-ES* [26]; 2) *DE* [52]; 3) *PSO* [53]; and 4) *GA* [54] are selected to solve each instance. Note that the selection of the four optimizers is biased. Due to the large number of existing EAs, we cannot take all of them into consideration. Here, our goal is simply to see whether insights into relationships between algorithm performance and variable interaction levels as calculated by MEE can be documented. Therefore, we limit the number of algorithm tested to four widely used EAs with default settings presented in Table V. Note that the maximal number of function evaluations used by each optimizer is always 10^5.

For each optimizer and each instance, 25 independent runs are conducted. The mean of the best solution found in the fixed number of function evaluations is recorded. For each function and each optimizer, the Spearman's rank correlation coefficient (r_s), Pearson correlation coefficient (r_p), and MIC (r_m) between DVI and the mean of best solutions found in the fixed number of function evaluations are calculated across the 21 instances. To complete the analysis, we also compare results from the MEE with EE and meta-model ($\bar{R}_l^2$ and $\bar{R}_q^2$). The statistic tests employed to determine the best performances are the same with Section IV-C.

V. ANALYSIS OF EXPERIMENTAL RESULTS

A. Identification of Pairwise Interaction (Q1)

1) Performances Comparison: Table VI lists the experimental results when MEE was used to identify pairwise interactions between decision variables in the 24 benchmark functions. The performances of MEE is compared with DG, PCC, and SRC in terms of *precision*, *recall*, and *F-score*. The different categories of the benchmark functions are divided by the double line. The best performances are highlighted in bold. We now provide a detailed analysis of the experimental results for each category.

Each of the methods, MEE, DG, and PCC, performs equally well on category I functions (see Section IV-A). Category I consists of four fully separable functions (f_1–f_4), each with ten separable decision variables. The MEE, DG, and PCC methods successfully identify all decision variables as being separable. Therefore, each method correctly records a score of 1.00 in terms of *precision*, *recall*, and *F-score*. The SRC method falsely classifies some separable decision variables as nonseparable, resulting in a low *precision* and *F-score*.

The MEE method achieves equal or better results than DG, PCC, or SRC on category II functions. Category II consists of four partially separable functions (f_5–f_8). Each function has five separable and nonseparable decision variables. The MEE method achieves the best score in terms of *precision*, *recall*, and *F-score* on f_5, f_6, and f_7. However on f_8, MEE identifies all the ten decision variables as being nonseparable, resulting in a low *precision*. The reason may be the computational error in the system. The DG and PCC methods correctly identify the five separable and nonseparable decision variables on f_6 and f_7. However on f_5, DG and PCC falsely classify some interacting decision variables as independent. On f_8, DG and PCC falsely classify some interacting variables as independent and some independent variables as interacting, resulting in a poor *precision* and *recall*. The SRC method identifies all the decision variables as being interacting, resulting in a low *precision* and *F-score*.

The MEE method achieves equal or better results than DG, PCC, or SRC on category III functions. Category III consists of four partially separable functions (f_9–f_{12}). Each function has five nonseparable subcomponents and each subcomponent has two decision variables. The MEE method accurately identifies all separable and nonseparable decision variables on each function. The DG and PCC methods achieve best score on f_{11} and f_{12}. However on f_9 and f_{10}, the performances of DG and PCC are worse than MEE in terms of *recall* and *F-score*. The SRC method is outperformed by the other methods on category III functions.

The MEE method achieves comparable or better results than DG on category IV functions. Category IV consists of four partially separable functions (f_{13}–f_{16}). Each function has two nonseparable subcomponents and each subcomponent has five decision variables. The MEE method achieves the best score for *precision*, *recall*, and *F-score* on f_{14}, f_{15}, and f_{16}. On f_{13}, MEE falsely identifies all decision variables as interacting, resulting in a poor *precision* (0.44). The DG method achieves the best *precision*, *recall*, and *F-score* on f_{15} and f_{16}. However, DG performs poorly on f_{13}, with *precision* $= 0.77$ and *recall* $= 0.82$ (1 is the best score). On f_{14}, DG sometimes fails to capture some interacting decision variables. The PCC and SRC methods perform worse than MEE on f_{13} and f_{16}.

The MEE method achieves better results than DG or PCC on category V functions. Category V consists of four overlapping functions (f_{17}–f_{20}). The MEE method achieves the best score

TABLE VI
EXPERIMENTAL RESULTS OF THE MEE, DG, PCC, AND SRC METHODS ON THE 24 BENCHMARK FUNCTIONS IN TERMS OF *Precision*, *Recall*, AND *F-Score*. THE MEAN RESULTS BASED ON 25 INDEPENDENT RUNS ARE PRESENTED. PCC AND SRC METHODS ARE EMPLOYED TO SHOW THE EFFICACY OF MIC COMPARED WITH PEARSON CORRELATION COEFFICIENT AND SPEARMAN'S RANK CORRELATION COEFFICIENT. THE BEST PERFORMANCES FROM THE FOUR METHODS ARE HIGHLIGHTED IN BOLD [WILCOXON RANK-SUM TESTS ($\alpha = 0.05$) WITH HOLM p-VALUE CORRECTION]

Cate	Func_ID	*Precision*				*Recall*				*F-Score*			
		MEE	DG	PCC	SRC	MEE	DG	PCC	SRC	MEE	DG	PCC	SRC
I	f_1	**1.00**	**1.00**	**1.00**	0.00	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.00
	f_2	**1.00**	**1.00**	**1.00**	0.00	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.00
	f_3	**1.00**	**1.00**	**1.00**	0.00	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.00
	f_4	**1.00**	**1.00**	**1.00**	0.00	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.00
II	f_5	**1.00**	**1.00**	**1.00**	0.22	**1.00**	0.95	0.30	**1.00**	**1.00**	0.97	0.46	0.36
	f_6	**1.00**	**1.00**	**1.00**	0.22	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.36
	f_7	**1.00**	**1.00**	**1.00**	0.22	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.36
	f_8	0.22	**0.24**	0.22	0.22	**1.00**	0.26	**1.00**	**1.00**	**0.36**	0.25	**0.36**	**0.36**
III	f_9	**1.00**	**1.00**	**1.00**	0.11	**1.00**	0.99	0.20	**1.00**	**1.00**	0.99	0.33	0.20
	f_{10}	**1.00**	**1.00**	**1.00**	0.11	**1.00**	0.99	0.20	**1.00**	**1.00**	0.99	0.33	0.20
	f_{11}	**1.00**	**1.00**	**1.00**	0.11	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.20
	f_{12}	**1.00**	**1.00**	**1.00**	0.11	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.20
IV	f_{13}	0.44	**0.77**	0.44	0.44	**1.00**	0.82	**1.00**	**1.00**	0.61	**0.79**	0.61	0.61
	f_{14}	**1.00**	**1.00**	**1.00**	0.44	**1.00**	0.95	0.35	**1.00**	**1.00**	0.97	0.51	0.61
	f_{15}	**1.00**	**1.00**	**1.00**	0.44	**1.00**	**1.00**	0.80	**1.00**	**1.00**	**1.00**	0.80	0.61
	f_{16}	**1.00**	**1.00**	**1.00**	0.44	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.61
V	f_{17}	1.00	1.00	1.00	1.00	**1.00**	0.11	**1.00**	0.99	**1.00**	0.20	**1.00**	0.99
	f_{18}	1.00	1.00	1.00	1.00	**1.00**	0.11	0.04	**1.00**	**1.00**	0.20	0.08	**1.00**
	f_{19}	1.00	1.00	1.00	1.00	0.98	0.38	0.33	**1.00**	0.99	0.55	0.50	**1.00**
	f_{20}	**1.00**	**1.00**	0.44	0.44	**1.00**	0.20	**1.00**	**1.00**	**1.00**	0.33	0.61	0.61
VI	f_{21}	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.93	**1.00**	**1.00**	**1.00**	0.95
	f_{22}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	f_{23}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	f_{24}	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	0.97	**1.00**	**1.00**	**1.00**	0.98

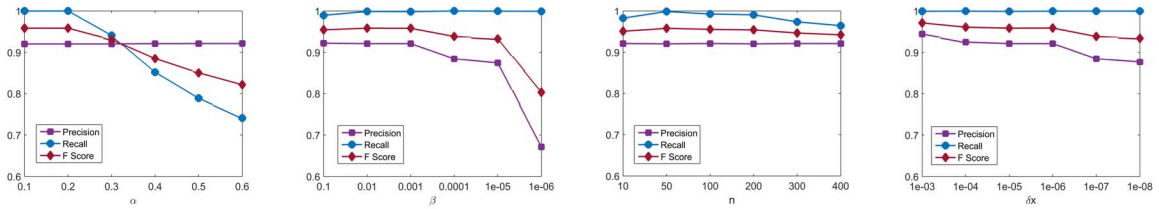

Fig. 2. Experimental results of the MEE method with different parameter settings on the 24 benchmark functions. Sensitivity to (a) α, (b) β, (c) n, and (d) δx. The horizontal axis represents the corresponding parameter values. The vertical axis represents the averaged *precision*, *recall*, and *F-score*, where "□" represents *precision*, "○" represents *recall*, and "◇" represents *F-score*.

on f_{17}, f_{18}, and f_{20} and nearly best score on f_{19} for *recall* and *F-score*. The DG method is unable to identify all variable interactions in this category. The *recall* rates of the DG method on the four overlapping functions are very low (less than 0.40), resulting in a low *F-score*. The *recall* rates of PCC on f_{18} and f_{19} are also very low. On f_{20}, the PCC method falsely classifies some independent variables as interacting, resulting in a low *precision*. The SRC method achieves comparable results with the MEE method on f_{17}, f_{18}, and f_{19}.

The MEE, DG, and PCC methods perform equally well on category VI functions. Category VI consists of four fully nonseparable functions (f_{21}–f_{24}), each with ten nonseparable decision variables. The MEE, DG, and PCC methods accurately identify all the decision variables as nonseparable. Therefore, all of them achieve the best score for *precision*, *recall*, and *F-score*. The SRC method performs slightly worse than the other methods.

In summary, the proposed MEE method achieves the best score (1.00) in terms of *precision*, *recall*, and *F-score* on 21 out of the 24 benchmark functions. MEE outperforms the DG, PCC, and SRC methods on the 24 benchmark functions. The comparison between MEE and PCC/SRC confirms that the MIC is more reliable than the Pearson correlation coefficient or Spearman's rank correlation coefficient when identifying functional relationships. The reason is that the Pearson correlation coefficient can only capture linear relationships and the Spearman's rank correlation coefficient can only capture increasing or decreasing relationships.

2) Parameter Sensitivity: The experimental results for the MEE method with different parameter settings on the 24 benchmark functions are presented in Fig. 2. Fig. 2(a)–(d) presents the sensitivity to α, β, n, and δx, respectively.

Fig. 2(a) shows that the MEE method performs well in a wide range of α value. In fact, when α is in the range of 0.1–0.3, *precision*, *recall*, and *F-score* are greater than 0.90. When α increases, *precision* increases slightly, while *recall* and *F-score* decreases. The reason is that when α increases, the threshold to identify variable interaction becomes higher.

Therefore, some pairwise interacting decision variables with small MIC value will be falsely identified as independent, resulting in a decrease in *recall*. On the other hand, some independent pairwise decision variables with large MIC value will be correctly identified as independent, resulting in an increase in *precision*. Consider two extreme cases: 1) if $\alpha = 0$, all pairwise decision variables will be identified as interacting, therefore *recall* = 1 and 2) if $\alpha = 1$, all pairwise decision variables will be identified as independent, therefore *recall* = 0, and *precision* = 1.

Fig. 2(b) shows that the MEE method performs well across a wide range of β values. When β is in the range of 10^{-3} to 10^{-1}, *precision*, *recall*, and *F-score* are greater than 0.90. When $\beta = 10^{-6}$, the *precision* and *F-score* decrease significantly. The reason may be that the computational error in the system is greater than 10^{-6}. Therefore, $\beta = 10^{-6}$ can no longer remove the computational errors from the system. This result suggests that the selection of β value should be greater than 10^{-6}.

Fig. 2(c) shows that the performance of the MEE method is not sensitive to the sample size n. When n is in the range of 10–400, *precision*, *recall*, and *F-score* are all greater than 0.92. Ideally, when n increases, *precision*, *recall*, and *F-score* should also increase. However, such phenomenon cannot be observed from Fig. 2(c) (the optimal sample size $n = 50$). So far, we have no explanation for this phenomenon.

Fig. 2(d) shows that the MEE method performs well when δx is in the range of 10^{-6} to 10^{-3} (*precision*, *recall*, and *F-score* greater than 0.90). The parameter δx is one of the main factors that result in computational errors in the system. When δx is less than 10^{-6}, *precision* and *F-score* decreases quickly. The reason may be that the computational errors increase significantly when $\delta x < 10^{-6}$. It suggests that the selection of δx should be greater than or equal to 10^{-6}.

The pairwise interactions between parameters are also investigated. Due to page limits, we do not present the detailed results in this paper. However, two conclusions can be drawn from the results: 1) no significant pairwise interactions can be observed between parameters except for $(\beta, \delta x)$ and 2) our proposed MEE method achieves high accuracy across a wide range of parameter settings when identifying variable interactions.

B. Quantification of Variable Interaction (Q2)

This section presents the experimental results of the proposed MEE measure when quantifying the level of variable interactions in the 24 benchmark functions. The performance of MEE is compared against the EE and meta-model.

As shown in Table VII, consistent results can be obtained by MEE for each category in most cases, while EE and meta-model ($\bar{R}_l^2$, $\bar{R}_q^2$) generate diverse results. In category I, MEE identifies all of the four functions as being fully separable (DVI = 0), while the results obtained by EE and meta-model are diverse: EE ranging from 0.22 to 2.01; $\bar{R}_l^2$ ranging from 0.00 to 0.25; $\bar{R}_q^2$ ranging from 0.59 to 1.00. In category II, MEE identifies all the functions as partially separable with DVI = 0.22 expect for f_8. On f_8, MEE unexpectedly identifies all decision variables as interacting. The reason may be that the fitness landscape of f_8 is highly rugged, resulting in large computational errors in the system. In category III, the MEE measure identifies all the functions as partially separable with DVI = 0.11, while we cannot observe any consistent results obtained by EE or meta-model. In category IV, the DVI obtained by MEE for f_{14}, f_{15}, and f_{16} are all 0.44, while for f_{13}, DVI = 1.00. The reason why MEE identifies all decision variables as being interacting in f_{13} is the same with f_8. In categories V and VI, MEE identifies all the functions as being fully nonseparable (DVI = 1.00). However, category V is fully nonseparable with indirect interaction (DIVI > 0), while category VI is fully nonseparable only with direct interaction (DIVI = 0). The results obtained by EE, $\bar{R}_l^2$ and $\bar{R}_q^2$ for categories V and VI are diverse. In summary, MEE is more robust and accurate when compared against EE and meta-model when quantifying variable interactions in a continuous optimization problem.

TABLE VII
EXPERIMENTAL RESULTS OF THE MEE, EE, AND META-MODEL ON THE 24 BENCHMARK FUNCTIONS IN TERMS OF QUANTIFYING VARIABLE INTERACTIONS. THE MEAN RESULTS BASED ON 25 INDEPENDENTS RUNS ARE PRESENTED. FOR EACH METHOD, CONSISTENT RESULTS ACROSS THE BENCHMARK FUNCTIONS IN EACH CATEGORY ARE HIGHLIGHTED IN BOLD (KRUSKAL–WALLIS ONE-WAY ANOVA TEST WITH $\alpha = 0.05$)

Cate	Func	EE	Meta-Model		MEE		
			$\bar{R}_l^2$	$\bar{R}_q^2$	DVI	DIVI	DDVI
I	f_1	0.29	0.00	1.00	**0.00**	**0.00**	**0.00**
	f_2	1.21	0.00	0.99	**0.00**	**0.00**	**0.00**
	f_3	2.01	0.25	0.90	**0.00**	**0.00**	**0.00**
	f_4	0.22	0.07	0.59	**0.00**	**0.00**	**0.00**
II	f_5	0.53	0.00	0.99	0.22	0.00	0.22
	f_6	1.38	0.23	0.93	0.22	0.00	0.22
	f_7	2.38	0.19	0.74	0.22	0.00	0.22
	f_8	0.82	0.06	0.28	1.00	0.29	0.71
III	f_9	0.05	0.00	0.05	**0.11**	**0.00**	**0.11**
	f_{10}	0.70	0.46	0.85	**0.11**	**0.00**	**0.11**
	f_{11}	0.57	0.25	0.89	**0.11**	**0.00**	**0.11**
	f_{12}	0.78	0.36	0.86	**0.11**	**0.00**	**0.11**
IV	f_{13}	1.25	0.18	0.46	1.00	0.25	0.75
	f_{14}	2.12	0.19	0.72	0.44	0.00	0.44
	f_{15}	0.60	0.21	0.73	0.44	0.00	0.44
	f_{16}	0.95	0.00	0.99	0.44	0.00	0.44
V	f_{17}	1.83	0.35	0.71	**1.00**	0.16	0.84
	f_{18}	0.54	0.15	0.54	**1.00**	0.80	0.20
	f_{19}	0.42	0.02	0.66	**1.00**	0.36	0.64
	f_{20}	1.53	0.36	0.74	**1.00**	0.38	0.62
VI	f_{21}	0.63	0.18	0.70	**1.00**	**0.00**	**1.00**
	f_{22}	0.72	0.05	0.28	**1.00**	**0.00**	**1.00**
	f_{23}	0.50	0.26	0.88	**1.00**	**0.00**	**1.00**
	f_{24}	0.76	0.16	0.50	**1.00**	**0.00**	**1.00**

C. Correlation With Algorithm Performances (Q3)

Fig. 3 shows the linear regression between the DVI and the mean of the best solutions obtained by each algorithm when solving the Elliptic/Rastrigin/Alpine benchmark functions. The corresponding Pearson correlation coefficient (r_p), Spearman's rank correlation coefficient (r_s), and MIC (r_m) values are listed in Table VIII.

High correlation can be observed between DVI and the mean of best solutions found by DE/GA/PSO on the three

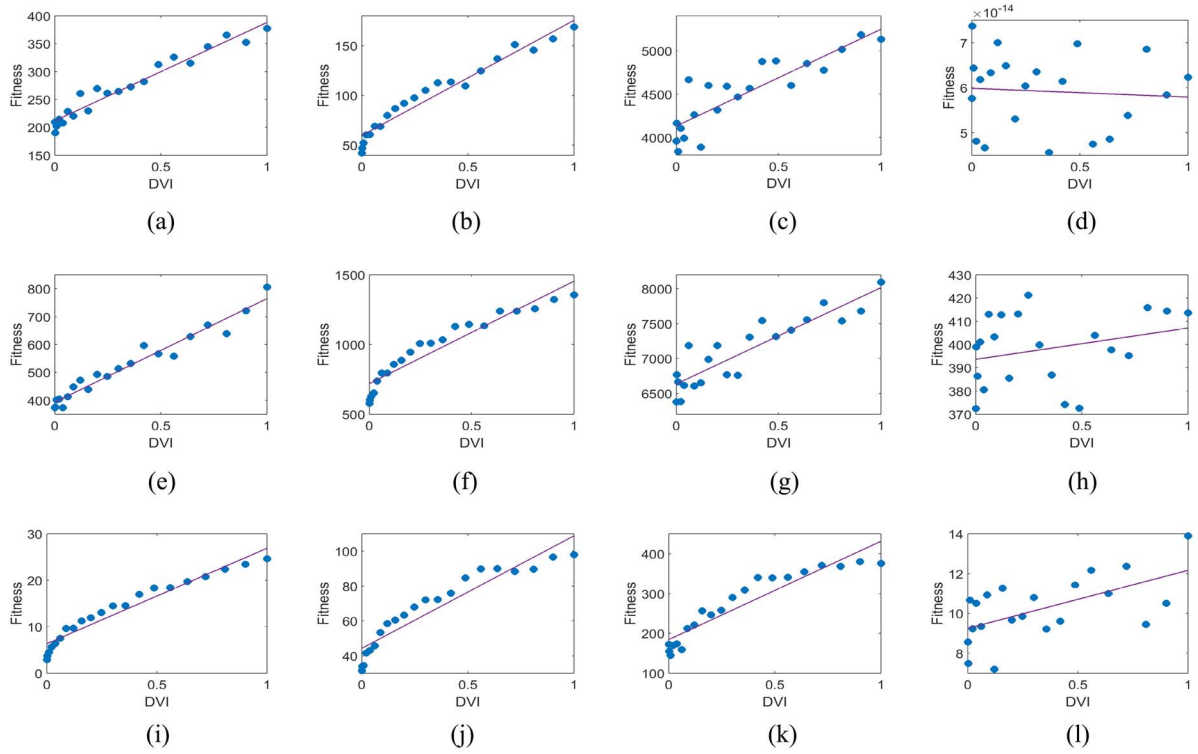

Fig. 3. Linear regression results for the DVI and the mean of the best solutions (fitness) found by each algorithm on each of the benchmark functions. Top: Elliptic function. Middle: Rastrigin function. Bottom: Alpine function. The vertical axis represents the mean of the best solutions found by the algorithm for a benchmark function. The horizontal axis represents the corresponding DVI returned by MEE. (a) DE_Ell. (b) GA_Ell. (c) PSO_Ell. (d) CMA-ES_Ell. (e) DE_Ras. (f) GA_Ras. (g) PSO_Ras. (h) CMA-ES_Ras. (i) DE_Alp. (j) GA_Alp. (k) PSO_Alp. (l) CMA-ES_Alp.

TABLE VIII
CORRELATION BETWEEN DVI/EE/$\bar{R}_l^2$/$\bar{R}_q^2$ AND THE MEAN OF THE BEST SOLUTIONS FOUND BY EACH OPTIMIZER ON EACH BENCHMARK FUNCTION. THE r_p REPRESENTS THE PEARSON CORRELATION COEFFICIENT; r_s REPRESENTS THE SPEARMAN'S RANK CORRELATION COEFFICIENT; r_m REPRESENTS THE MIC. THE HIGHEST CORRELATION (ABSOLUTE VALUE) IS HIGHLIGHTED IN BOLD

Functions	Measures	DE			GA			PSO			CMA-ES		
		r_p	r_s	r_m	r_p	r_s	r_m	r_p	r_s	r_m	r_p	r_s	r_m
Elliptic	DVI	**0.97**	**0.98**	**0.91**	**0.97**	**0.99**	**0.99**	**0.87**	**0.86**	**0.76**	-0.07	-0.11	0.19
	EE	0.38	0.29	0.37	0.50	0.37	0.40	0.46	0.29	0.37	**-0.13**	-0.08	0.24
	$\bar{R}_l^2$	0.32	0.36	0.38	0.35	0.38	0.38	0.47	0.48	0.38	**-0.13**	**-0.15**	0.24
	$\bar{R}_q^2$	-0.92	**-0.98**	**0.91**	-0.96	-0.98	**0.99**	-0.86	-0.85	**0.76**	0.11	0.10	**0.30**
Rastrigin	DVI	**0.97**	**0.97**	0.90	**0.95**	**0.99**	**0.99**	**0.90**	**0.88**	0.76	0.28	0.35	0.33
	EE	0.64	0.62	0.52	0.67	0.63	0.50	0.67	0.57	0.57	0.38	**0.46**	0.36
	$\bar{R}_l^2$	-0.81	-0.89	0.91	-0.86	-0.89	0.91	-0.76	-0.75	0.76	**-0.41**	-0.40	**0.41**
	$\bar{R}_q^2$	-0.67	-0.93	**0.99**	-0.81	-0.95	**0.99**	-0.67	-0.85	**0.81**	-0.36	-0.27	0.26
Alpine	DVI	0.96	**1.00**	**0.99**	0.94	**0.98**	**0.99**	0.93	**0.97**	**0.99**	**0.59**	**0.56**	0.37
	EE	0.58	0.57	0.50	0.54	0.58	0.50	0.46	0.48	0.37	0.22	0.02	0.31
	$\bar{R}_l^2$	0.87	0.87	0.86	0.87	0.85	0.84	0.85	0.84	0.84	0.37	0.42	**0.50**
	$\bar{R}_q^2$	**-0.98**	-0.99	**0.99**	**-0.98**	-0.97	**0.99**	**-0.96**	-0.96	**0.99**	-0.58	-0.55	0.37

benchmark functions. In fact, the r_p, r_s, and r_m are close to 1 in most cases. The plot shows that when DVI increases from 0 to 1, the optimization problem becomes more difficult for DE/GA/PSO to solve.

Take DE as an example, there is a strictly increasing relationship between DVI and the mean of best solutions found on Alpine benchmarks [Fig. 3(i)]. The easiest function is the one with DVI $= 0$, and the hardest function is the one with DVI $= 1$. On the Elliptic and Rastrigin functions, there is a high correlation between DVI and the mean of best solutions found ($r_s = 0.98/0.97$). Similar conclusions can be drawn for GA and PSO. However for CMA-ES, there is no significant correlation between DVI and the mean of best solutions found on Elliptic/Rastrigin benchmarks ($r_s = -0.11/0.35$). The reason for this may be that CMA-ES is designed to be rotationally invariant. Therefore, it performs equally well on separable and nonseparable functions [55].

We cannot observe significant correlation between EE and the mean of the best solutions found by DE/GA/PSO/CMA-ES on the three benchmark functions. Therefore, EE may not be a reliable measure in terms of quantifying interactions between continuous variables.

The correlations between $\bar{R}_l^2$ and the mean of the best solutions found by DE/GA/PSO on Rastrigin and Alpine benchmark functions are generally significant. However paradoxically, on the Rastrigin function the correlation is negative

($r_s < 0$), while on Alpine function the correlation is positive ($r_s > 0$).

The $\bar{R}_q^2$ measure performs well on the three benchmark functions. Note that the correlation between $\bar{R}_q^2$ and the mean of the best solutions found by each optimizer should be negative ($r_s < 0$). The reason for this is that as the DVI increases, an optimization problem becomes more complex, therefore the model accuracy $\bar{R}_q^2$ decreases. However, we observe that the model accuracy $\bar{R}_q^2$ varies significantly across different benchmark functions with the same DVIs. For example the model accuracy $\bar{R}_q^2 = 0.99$ on fully separable Elliptic function; while $\bar{R}_q^2 = 0.48$ on fully separable Alpine function. The reason for this may be that, apart from variable interactions, other important problem characteristics, e.g., modality also have significant impacts on the model accuracy $\bar{R}_q^2$ [5]. This observation is consistent with the results shown in Table VII.

VI. Algorithm Design Using MEE

In this section, we investigate how the DVIs within a continuous optimization problem can be used to determine the choice of algorithms used. Here, we examine the overall optimization performance when the MEE measure is embedded into an algorithm design framework.

The separability prototype for automatic memes (SPAM) [21] framework consists of two stages: a separability analysis stage and an optimization stage. In the first stage, the level of variable interactions is approximated. Then the SPAM framework selects appropriate operator(s) to guide the search based on the level of variable interactions. In the original paper [21], two evolutionary operators are employed—the *S* operator [56] and the *R* operator [57]. The *S* operator searches along each dimension, which is efficient to solve separable problems. The *R* operator (which is the Rosenbrock algorithm) perturbs all the variables simultaneously to follow the gradient of the fitness landscape. The *R* operator is designed for nonseparable optimization problems. In this paper, we use the same candidate operators (*R* and/or *S*) used by Caraffini *et al.* [21]. Note that other efficient algorithms for separable or nonseparable problems can also be used as the candidate operators.

In the original SPAM framework, CMA-ES is employed to obtain the covariance matrix (C) within a given computational budget. Then the correlation coefficient between each pair of decision variables is calculated

$$\rho_{i,j} = \frac{C_{i,j}}{\sqrt{C_{i,i}C_{j,j}}}. \tag{19}$$

The $|\rho_{i,j}|$ is normalized to 0 if $0 \leq |\rho_{i,j}| < 0.2$, to 0.3 if $0.2 \leq |\rho_{i,j}| < 0.4$, to 0.5 if $0.4 \leq |\rho_{i,j}| < 0.6$, to 0.7 if $0.6 \leq |\rho_{i,j}| < 0.8$, to 1 if $0.8 \leq |\rho_{i,j}| \leq 1$. Then an index ξ is calculated based on the normalized matrix ρ^n as the approximation of the level of variable interactions

$$\xi = \frac{2}{d(d-1)} \sum_{i=1}^{d-1} \sum_{j=i+1}^{d} \rho_{i,j}^n \tag{20}$$

where d is the dimensionality. The index ξ is then employed to calculate the activation probabilities of the *S* and *R* operators. Note that the best solution found by CMA-ES in this stage (separability analysis stage) will be passed to the following stage as the initial point.

In the subsequent optimization stage, the index ξ is used to assign an activation probability to each operator. If $\xi = 0$, the optimization is fully separable, therefore only the *S* operator is used to solve the problem. If $\xi \geq 0.5$, the optimization problem is regarded as fully nonseparable and only the *R* operator is used to guide the search. If $0 < \xi < 0.5$, the optimization problem is considered as partially separable and two operators coexist with activation probabilities defined as follows:

$$p_s = 1 - 2\xi, \quad p_r = 2\xi. \tag{21}$$

The original SPAM framework has been tested on multiple benchmark suites and achieved good results compared with the state-of-the-art optimization algorithms [21]. However, the framework does have some limitations.

1) In the original SPAM framework, the Pearson correlation coefficient is used to identify variable interactions, which is not reliable as we have shown in Section V-A.
2) Normalizing the coefficient matrix $|\rho|$ to $|\rho^d|$ adds bias.
3) It is not reasonable to consider the optimization problem as fully nonseparable when $\xi \geq 0.5$.

To address these limitations, the MEE measure can be used to quantify the level of variable interactions, and the DVI value generated by MEE can be used to assign the activation probabilities to the *S* and *R* operators

$$p_s = 1 - \text{DVI}, \quad p_r = \text{DVI}. \tag{22}$$

Given the discussion above, we test this new version of SPAM framework where MEE is embedded. We denote our model as SPAM_MEE. Note that the best solution found in the separability analysis stage (MEE stage) will also be passed to the optimization stage.

The results obtained by SPAM_MEE are compared against the results generated using SPAM and SPAM0.5 on the 24 benchmark functions (Table IX). SPAM0.5 is a version of SPAM with the activation probabilities of the *S* and *R* operators both equal to 0.5. To save computational cost in the separability analysis stage, the sample size used in MEE was set to $n = 10$, while other parameters are the default values highlighted in Table IV. Therefore the number of function evaluations used by MEE is $10d(d-1)$. The maximum number of function evaluations was set to $3 \times 10^3 d$, divided between the separability analysis stage and optimization stage. Other parameter setting is consistent with [21]. To complete the analysis, we also compare SPAM_MEE with CMA-ES [26]. The parameter setting for CMA-ES is consistent with the one presented in Table V, but the maximum number of function evaluations was set to $3 \times 10^3 d$.

For each algorithm and benchmark function combination, 25 independent runs were carried out. The mean and standard deviation of the best solutions found within the given computational budget are recorded and shown in Table IX. The two-sided Wilcoxon test ($\alpha = 0.05$) with Holm p-value correction [51] was used to determine the best performance from SPAM_MEE, SPAM and SPAM0.5 in a pairwise fashion. The best performances are highlighted in bold in Table IX. We

TABLE IX
EXPERIMENTAL RESULTS OF SPAM_MEE, SPAM, AND SPAM0.5 ON THE 24 BENCHMARK FUNCTIONS. THE BEST PERFORMANCES OF THE THREE ALGORITHMS ARE HIGHLIGHTED IN BOLD. THE PERFORMANCE OF SPAM_MEE IS ALSO COMPARED WITH CMA-ES. THE BETTER RESULTS OBTAINED BY SPAM_MEE AND CMA-ES IS MARKED WITH † (TWO-SIDED WILCOXON RANK-SUM TEST WITH THE CONFIDENCE INTERVAL OF 95%)

Func Num	SPAM_MEE		SPAM		SPAM0.5		CMA-ES	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
f_1	6.89e-62	3.35e-62	8.60e-62	8.38e-62	8.60e-62	7.92e-62	8.19e-62	7.46e-62
f_2	5.68e-60†	1.01e-59	5.79e-60	9.16e-60	5.79e-60	8.58e-60	2.04e-59	7.82e-59
f_3	**1.04e+02**	4.38e+01	1.49e+02	7.28e+01	1.30e+02	5.63e+01	2.11e+01†	1.20e+01
f_4	**3.34e-14**†	2.59e-14	1.24e-13	1.33e-13	1.66e-13	1.25e-13	5.50e-13	1.92e-12
f_5	**1.27e+01**	4.81e+00	1.55e+01	4.26e+00	1.46e+01	4.72e+00	1.22e+01	5.85e+00
f_6	1.80e-45	8.74e-45	1.19e-45	2.76e-45	6.60e-39	3.30e-38	8.16e-47	2.28e-46
f_7	**9.98e+01**	4.73e+01	1.69e+02	1.01e+02	1.31e+02	9.15e+01	1.14e+02	6.70e+01
f_8	9.30e+00	3.77e+00	7.92e+00	3.32e+00	8.23e+00	3.84e+00	8.46e+00	4.57e+00
f_9	**7.73e-01**†	1.83e-01	1.44e+00	4.21e-01	1.46e+00	3.64e-01	1.32e+00	3.24e-01
f_{10}	**1.36e+02**	1.40e+02	4.07e+03	4.42e+03	3.76e+03	3.28e+03	1.21e+01†	1.11e+01
f_{11}	6.33e-01	2.48e-01	6.35e-01	3.12e-01	6.92e-01	2.54e-01	5.25e-01	2.76e-01
f_{12}	**1.39e-01**†	5.60e-01	4.71e-01	1.06e+00	7.68e-01	1.83e+00	3.96e-01	1.21e+00
f_{13}	1.41e+01	7.78e+00	1.21e+01	4.48e+00	1.32e+01	5.45e+00	1.17e+01	8.34e+00
f_{14}	2.00e-01†	1.09e-01	2.80e-01	2.06e-01	3.80e-01	2.46e-01	3.63e-01	2.17e-01
f_{15}	8.09e+01	4.09e+01	9.44e+01	7.53e+01	7.48e+01	5.41e+01	1.58e+01†	1.25e+01
f_{16}	**3.06e+00**	2.57e+00	3.85e+00	4.32e+00	3.19e+00	1.71e+00	2.35e-12†	6.40e-12
f_{17}	**4.55e+00**	1.73e+01	3.58e+01	1.08e+02	5.60e+01	1.21e+02	2.02e-01†	8.00e-01
f_{18}	**3.08e+00**†	6.74e-01	4.37e+00	1.11e+00	4.46e+00	1.30e+00	4.12e+00	1.20e+00
f_{19}	**2.73e+01**†	2.17e+01	7.38e+01	3.52e+01	8.26e+01	2.35e+01	4.77e+01	4.53e+01
f_{20}	**2.86e+00**	9.43e+00	4.99e+01	8.18e+01	2.80e+02	5.89e+02	1.33e+00	2.24e+00
f_{21}	**1.55e+02**	6.81e+01	2.30e+02	1.31e+02	3.17e+02	1.83e+02	2.98e+01†	1.86e+01
f_{22}	5.38e+01	1.22e+01	5.17e+01	1.05e+01	5.43e+01	1.52e+01	4.73e+01	1.43e+01
f_{23}	**3.24e-18**	4.19e-18	5.32e-18	1.35e-17	2.15e-15	1.07e-14	2.05e-18†	5.88e-18
f_{24}	3.54e+02	2.22e+02	3.49e+02	2.24e+02	3.25e+02	1.68e+02	2.97e+02	1.63e+02

observed that SPAM_MEE achieves equal or better results than SPAM and SPAM0.5 on the 24 benchmark functions. Note that SPAM_MEE uses more function evaluations than SPAM or SPAM0.5 in the separability analysis stage. Therefore, the number of function evaluations used by SPAM_MEE in the optimization stage is less than that used by SPAM or SPAM0.5. This finding suggests that it may be worth assigning a portion of computational budget to gain an insight into the fitness landscape of an optimization problem. The information, in turn, may guide the search toward better region of the fitness landscape. The SPAM_MEE achieves comparable results with CMA-ES when solving the 24 benchmark functions.

VII. CONCLUSION

In this paper, we have investigated the effects of interactions between decision variables in continuous optimization problems. Here, the overarching goal was to examine whether it was possible to improve the performance of an optimization algorithm by guiding the search based on the level of variable interactions. A robust ELA measure for continuous optimization problems, MEE, was introduced. MEE identifies the IM of decision variables in a continuous optimization problem, which is then used to generate an accurate measure of the DVI (encapsulating both direct and indirect variable interactions) spanning a suite of benchmark problems. We have shown that the solution quality found by an EA is correlated with the level of variable interaction in a given problem. This observation suggests that fitness landscape characteristic captured by MEE can be used as a source of information to predict algorithm performance. Significantly, when the MEE measure was embedded into an optimization algorithm design framework, the performance of our model was at least as good, and in many cases outperformed, the SPAM model on the suite of benchmark problems. These results confirm our hypothesis that quantifying the level of variable interactions can be used to guide the search toward better regions of the fitness landscapes.

In future work, we plan to complete further analysis of the MEE model and explore additional avenues to guide the search using the DVIs. Another direction worth pursuing is focussed on the development of an algorithm selection framework, where a range of ELA measure are combined with MEE, as described by Muñoz *et al.* [10], [12].

ACKNOWLEDGMENT

The authors would like to thank X. Li, M. A. Muñoz, and W. Wang for their valuable comments.

REFERENCES

[1] L. Kallel, B. Naudts, and C. R. Reeves, "Properties of fitness functions and search landscapes," *Theor. Aspects Evol. Comput.*, pp. 175–206, 2001.

[2] T. Weise, R. Chiong, and K. Tang, "Evolutionary optimization: Pitfalls and booby traps," *J. Comput. Sci. Technol.*, vol. 27, no. 5, pp. 907–936, Nov. 2012.

[3] M. N. Omidvar, X. Li, and K. Tang, "Designing benchmark problems for large-scale continuous optimization," *Inf. Sci.*, vol. 316, pp. 419–436, Sep. 2015.

[4] O. Mersmann, M. Preuss, and H. Trautmann, *Benchmarking Evolutionary Algorithms: Towards Exploratory Landscape Analysis*. Berlin, Germany: Springer, 2010, pp. 73–82.

[5] O. Mersmann *et al.*, "Exploratory landscape analysis," in *Proc. 13th Annu. Conf. Genet. Evol. Comput.*, Dublin, Ireland, 2011, pp. 829–836.

[6] O. Mersmann, M. Preuss, H. Trautmann, B. Bischl, and C. Weihs, "Analyzing the BBOB results by means of benchmarking concepts," *Evol. Comput.*, vol. 23, no. 1, pp. 161–185, 2015.

[7] P. F. Stadler, "Fitness landscapes," *Biol. Evol. Stat. Phys.*, vol. 585, pp. 183–204, 2002.

[8] T. Jones and S. Forrest, "Fitness distance correlation as a measure of problem difficulty for genetic algorithms," in *Proc. ICGA*, vol. 95. Pittsburgh, PA, USA, 1995, pp. 184–192.

[9] K. M. Malan and A. P. Engelbrecht, "A survey of techniques for characterising fitness landscapes and some possible ways forward," *Inf. Sci.*, vol. 241, pp. 148–163, Aug. 2013.

[10] M. A. Muñoz, M. Kirley, and S. K. Halgamuge, "The algorithm selection problem on the continuous optimization domain," *Comput. Intell. Intell. Data Anal.*, vol. 445, pp. 75–89, 2013.

[11] M. A. Muñoz, M. Kirley, and S. K. Halgamuge, "Exploratory landscape analysis of continuous space optimization problems using information content," *IEEE Trans. Evol. Comput.*, vol. 19, no. 1, pp. 74–87, Feb. 2015.

[12] M. A. Muñoz, Y. Sun, M. Kirley, and S. K. Halgamuge, "Algorithm selection for black-box continuous optimization problems: A survey on methods and challenges," *Inf. Sci.*, vol. 317, pp. 224–245, Oct. 2015.

[13] Y. Davidor, "Epistasis variance: A viewpoint on GA-hardness," *Found. Genet. Algorithms*, vol. 1, pp. 23–35, 1991.

[14] S. Rochet, G. Venturini, M. Slimane, and E. M. E. Kharoubi, "A critical and empirical study of epistasis measures for predicting GA performances: A summary," *Artif. Evol.*, vol. 1363, pp. 275–285, 1998.

[15] B. Naudts and L. Kallel, "A comparison of predictive measures of problem difficulty in evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 4, no. 1, pp. 1–15, Apr. 2000.

[16] C. R. Reeves and C. Wright, "An experimental design perspective on genetic algorithms," in *Proc. Found. Genet. Algorithms (FOGA)*, San Mateo, CA, USA, 1995, pp. 7–22.

[17] C. Fonlupt, D. Robilliard, and P. Preux, "A bit-wise epistasis measure for binary search spaces," in *Parallel Problem Solving From Nature—PPSN V*. Heidelberg, Germany: 1998, pp. 47–56.

[18] D.-I. Seo and B.-R. Moon, "An information-theoretic analysis on the interactions of variables in combinatorial optimization problems," *Evol. Comput.*, vol. 15, no. 2, pp. 169–198, 2007.

[19] D. N. Reshef *et al.*, "Detecting novel associations in large data sets," *Science*, vol. 334, no. 6062, pp. 1518–1524, 2011.

[20] D. Albanese *et al.*, "Minerva and minepy: A C engine for the MINE suite and its R, Python and MATLAB wrappers," *Bioinformatics*, vol. 29, no. 3, pp. 407–408, 2013.

[21] F. Caraffini, F. Neri, and L. Picinali, "An analysis on separability for memetic computing automatic design," *Inf. Sci.*, vol. 265, pp. 1–22, May 2014.

[22] T. Ray and X. Yao, "A cooperative coevolutionary algorithm with correlation based adaptive variable partitioning," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Trondheim, Norway, 2009, pp. 983–989.

[23] J. Smith and T. C. Fogarty, "An adaptive poly-parental recombination strategy," *Evol. Comput.*, vol. 993, pp. 48–61, 1995.

[24] G. R. Harik, "Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms," Ph.D. dissertation, Dept. Comput. Sci. Eng., Univ. Michigan, Ann Arbor, MI, USA, 1997.

[25] H. Mühlenbein, J. Bendisch, and H.-M. Voigt, "From recombination of genes to the estimation of distributions II. Continuous parameters," in *Parallel Problem Solving From Nature—PPSN IV*. Heidelberg, Germany: Springer, 1996, pp. 188–197.

[26] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evol. Comput.*, vol. 11, no. 1, pp. 1–18, 2003.

[27] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 378–393, Jun. 2014.

[28] M. Munetomo and D. E. Goldberg, "Linkage identification by non-monotonicity detection for overlapping functions," *Evol. Comput.*, vol. 7, no. 4, pp. 377–398, 1999.

[29] Y. Sun, S. K. Halgamuge, M. Kirley, and M. A. Muñoz, "On the selection of fitness landscape analysis metrics for continuous optimization problems," in *Proc. 7th Int. Conf. Inf. Autom. Sustain. (ICIAfS)*, Colombo, Sri Lanka, 2014, pp. 1–6.

[30] M. Lunacek and D. Whitley, "The dispersion metric and the CMA evolution strategy," in *Proc. 8th Annu. Conf. Genet. Evol. Comput.*, Seattle, WA, USA, 2006, pp. 477–484.

[31] Y. Sun, M. Kirley, and S. K. Halgamuge, "Extended differential grouping for large scale global optimization with direct and indirect variable interactions," in *Proc. Genet. Evol. Comput. Conf.*, Madrid, Spain, 2015, pp. 313–320.

[32] R. T. Rockafellar and R. J. B. Wets, *Variational Analysis*, vol. 317. Heidelberg, Germany: Springer 2009.

[33] B. S. Mordukhovich, N. M. Nam, and N. T. Y. Nhi, "Partial second-order subdifferentials in variational analysis and optimization," *Numer. Funct. Anal. Optim.*, vol. 35, nos. 7–9, pp. 1113–1151, 2014.

[34] F. J. Poelwijk, D. J. Kiviet, D. M. Weinreich, and S. J. Tans, "Empirical fitness landscapes reveal accessible evolutionary paths," *Nature*, vol. 445, no. 7126, pp. 383–386, 2007.

[35] E. Weinberger, "Correlated and uncorrelated fitness landscapes and how to tell the difference," *Biol. Cybern.*, vol. 63, no. 5, pp. 325–336, 1990.

[36] D. Stowell and M. D. Plumbley, "Fast multidimensional entropy estimation by κ-d partitioning," *IEEE Signal Process. Lett.*, vol. 16, no. 6, pp. 537–540, Jun. 2009.

[37] S. B. Green, "How many subjects does it take to do a regression analysis," *Multivar. Behav. Res.*, vol. 26, no. 3, pp. 499–510, 1991.

[38] S. Verel, G. Ochoa, and M. Tomassini, "Local optima networks of NK landscapes with neutrality," *IEEE Trans. Evol. Comput.*, vol. 15, no. 6, pp. 783–797, Dec. 2011.

[39] P. Caamaño, F. Bellas, J. A. Becerra, and R. J. Duro, "Evolutionary algorithm characterization in real parameter optimization problems," *Appl. Soft Comput.*, vol. 13, no. 4, pp. 1902–1921, 2013.

[40] N. Manukyan, M. J. Eppstein, and J. S. Buzas, "Tunably rugged landscapes with known maximum and minimum," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 263–274, Apr. 2016.

[41] J. He, T. Chen, and X. Yao, "On the easiest and hardest fitness functions," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 295–305, Apr. 2015.

[42] J. Buzas and J. Dinitz, "An analysis of $N!K$ landscapes: Interaction structure, statistical properties, and expected number of local optima," *IEEE Trans. Evol. Comput.*, vol. 18, no. 6, pp. 807–818, Dec. 2014.

[43] P. Kerschke, M. Preuss, S. Wessing, and H. Trautmann, "Detecting funnel structures by means of exploratory landscape analysis," in *Proc. Genet. Evol. Comput. Conf.*, Madrid, Spain, 2015, pp. 265–272.

[44] T.-L. Yu, D. E. Goldberg, K. Sastry, C. F. Lima, and M. Pelikan, "Dependency structure matrix, genetic algorithms, and effective recombination," *Evol. Comput.*, vol. 17, no. 4, pp. 595–626, 2009.

[45] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Inf. Sci.*, vol. 178, no. 15, pp. 2985–2999, 2008.

[46] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948.

[47] T. Bossomaier, L. Barnett, and M. Harré, "Information and phase transitions in socio-economic systems," *Complex Adap. Syst. Model.*, vol. 1, no. 9, pp. 1–25, 2013.

[48] T. Speed, "A correlation for the 21st century," *Science*, vol. 334, no. 6062, pp. 1502–1503, 2011.

[49] J. Walters-Williams and Y. Li, "Estimation of mutual information: A survey," in *Rough Sets and Knowledge Technology*. Heidelberg, Germany: Springer, 2009, pp. 389–396.

[50] X. Li *et al.*, "Benchmark functions for the CEC'2013 special session and competition on large-scale global optimization," *Gene*, vol. 7, no. 33, p. 8, 2013.

[51] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*. Boca Raton, FL, USA: CRC Press, 2003.

[52] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[53] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Piscataway, NJ, USA, 1995, pp. 1942–1948.

[54] D. E. Golberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA, USA: Addison-Wesley, 1989.

[55] N. Hansen, R. Ros, N. Mauny, M. Schoenauer, and A. Auger, "Impacts of invariance in search: When CMA-ES and PSO face ill-conditioned and non-separable problems," *Appl. Soft Comput. J.*, vol. 11, no. 8, pp. 5755–5769, 2011.

[56] L.-Y. Tseng and C. Chen, "Multiple trajectory search for large scale global optimization," in *Proc. IEEE Congr. Evol. Comput.*, Hong Kong, 2008, pp. 3052–3058.

[57] H. H. Rosenbrock, "An automatic method for finding the greatest or least value of a function," *Comput. J.*, vol. 3, no. 3, pp. 175–184, 1960.

Yuan Sun (S'14) received the B.Sc. degree in theoretical and applied mechanics from Peking University, Beijing, China, in 2013. He is currently pursuing the Ph.D. degree with the University of Melbourne, Parkville, VIC, Australia, with a focus on fitness landscape analysis for continuous optimization problems, and its application in empirical models of algorithm performance.

His current research interests include exploratory landscape analysis, large scale optimization, multi-objective optimization, and feature selection using evolutionary computation approaches.

Michael Kirley (M'01) received the B.Ed. degree from Deakin University, Geelong, VIC, Australia, in 1998, and the Ph.D. degree from Charles Sturt University, Manly, NSW, Australia, in 2003.

He is currently an Associate Professor with the Department of Computing and Information Systems, University of Melbourne, Parkville, VIC, Australia. His current research interests include theory and application of evolutionary computation, evolutionary game theory, and complex systems science. He has published over 100 papers in the above areas.

Saman K. Halgamuge (SM'85) received the B.Sc.Eng. degree in electronic and telecommunication engineering from the University of Moratuwa, Moratuwa, Sri Lanka, and the Dr.-Ing. and Dipl.-Ing. degrees in electrical engineering (data engineering) from TU Darmstadt, Darmstadt, Germany, in 1995 and 1990, respectively.

He is a Professor and the Director of the Research School of Engineering, Australian National University, Canberra, ACT, Australia. He held appointments as a Professor with the Department of Mechanical Engineering and the Associate Dean of the International of Melbourne School of Engineering, University of Melbourne, Parkville, VIC, Australia. He is also the Professor V. K. Samaranayake Endowed Visiting Chair with the University of Colombo, Colombo, Sri Lanka. His current research interests include data engineering, optimization, smart grids, mechatronics, and bioinformatics. He has published over 250 papers and graduated 32 Ph.D. students in the above areas and obtained substantial research grants from the ARC, National Health and Medical Research Council, and industry.