Objective-Domain Dual Decomposition: An Effective Approach to Optimizing Partially Differentiable Objective Functions

Yiu-ming Cheung, *Fellow, IEEE*, Fangqing Gu, Hai-Lin Liu, Kay Chen Tan, *Fellow, IEEE*, and Han Huang

***Abstract*—This paper addresses a class of optimization problems in which either part of the objective function is differentiable while the rest is nondifferentiable or the objective function is differentiable in only part of the domain. Accordingly, we propose a dual-decomposition-based approach that includes both objective decomposition and domain decomposition. In the former, the original objective function is decomposed into several relatively simple subobjectives to isolate the nondifferentiable part of the objective function, and the problem is consequently formulated as a multiobjective optimization problem (MOP). In the latter decomposition, we decompose the domain into two subdomains, that is, the differentiable and nondifferentiable domains, to isolate the nondifferentiable domain of the nondifferentiable subobjective. Subsequently, the problem can be optimized with different schemes in the different subdomains. We propose a population-based optimization algorithm, called the simulated water-stream algorithm (SWA), for solving this MOP. The SWA is inspired by the natural phenomenon of water streams moving toward a basin, which is analogous to the process of searching for the minimal solutions of an optimization problem. The proposed SWA combines the deterministic search and heuristic search in a single framework. Experiments show that the SWA yields promising results compared with its existing counterparts.**

***Index Terms*—Domain decomposition, hybrid process, objective decomposition, partial differentiable objective function, simulated water-stream algorithm (SWA).**

I. Introduction

OPTIMIZATION problems are quite common in a variety of scientific areas. Without loss of generality, an optimization problem can be defined as follows:

$$\min_{\mathbf{x}\in D} F(\mathbf{x}) = (F_1(\mathbf{x}), \ldots, F_k(\mathbf{x}))^{\mathrm{T}}. \quad (1)$$

Here, $F_i(\mathbf{x})$ is the ith objective of the optimization problem and k is the number of objectives. $\mathbf{x} = (x_1, \ldots, x_n)^{\mathrm{T}} \in D$ is the decision variable vector, where $D = \prod_{i=1}^{n} [a_i, b_i] \subset R^n$ is the domain of the decision variable vector and n is its dimensionality. T denotes the transpose of a vector, and a_i and b_i are the upper and lower bounds, respectively, on the ith component x_i of $\mathbf{x}$. If $k = 1$, the problem is a single-objective optimization problem (SOP); otherwise, it is a multiobjective optimization problem (MOP).

Optimization problems are widely encountered in various applications. Among them, in a certain class of optimization problems, the objective function is nondifferentiable as a whole. Nevertheless, some number of the subobjective functions or part of one subobjective function may be differentiable. Furthermore, a nondifferentiable subobjective function may be nondifferentiable over the entire domain but differentiable in a subdomain. We call such problems partially differentiable problems. Instances of such problems in the literature include variable selection and feature extraction in machine learning [1], [2]; sparse representation in signal processing [3], [4]; and sparse autoencoder neural networks [5], to name a few.

Over the past decades, researchers have developed a number of optimization algorithms in the literature. These algorithms can be divided into two basic categories. The first one consists of differential-based optimization algorithms, such as the conjugate gradient method [6], the steepest descent method, and the quasi-Newton method [7]. In general, these algorithms can quickly obtain a globally optimal solution for a differentiable convex optimization problem [8]. However, they almost converge to a locally optimal solution when solving the multimodal optimization problems. Recently, researchers have developed some global optimization algorithms for multimodal SOPs, such as the tunneling algorithm [9], [10] and the filled function method [11]–[13]. Experimental studies have demonstrated the effectiveness of these algorithms in their application domains. Nevertheless, these algorithms cannot guarantee convergence to a globally optimal solution for

Manuscript received April 8, 2018; revised July 16, 2018; accepted September 11, 2018. Date of publication October 8, 2018; date of current version January 21, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61672444, Grant 61272366, and Grant 61673121, in part by the Natural Science Foundation of SZU under Grant 2017078, in part by the Faculty Research Grant of Hong Kong Baptist University (HKBU) under Project FRG2/17-18/082 and Project FRG2/16-17/051, in part by the KTO Grant of HKBU under Project MPCF-004-2017/18, in part by the SZSTI under Grant JCYJ20160531194006833, in part by the Projects of Science and Technology of Guangzhou under Grant 201508010008, and in part by the Natural Science Foundation of Guangdong Province under Grant 2017A030310467. This paper was recommended by Associate Editor G. G. Yen. *(Corresponding authors: Yiu-ming Cheung; Fangqing Gu.)*

Y.-M. Cheung is with the Department of Computer Science, Hong Kong Baptist University, Hong Kong (e-mail: ymc@comp.hkbu.edu.hk).

F. Gu and H.-L. Liu are with the School of Applied Mathematics, Guangdong University of Technology, Guangdong 510520, China (e-mail: fqgu@gdut.edu.cn; hlliu@gdut.edu.cn).

K. C. Tan is with the Department of Computer Science, City University of Hong Kong, Hong Kong (e-mail: kaytan@cityu.edu.hk).

H. Huang is with the School of Software Engineering, South China University of Technology, Guangzhou 510006, China (e-mail: hhan@scut.edu.cn).

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/TCYB.2018.2870487

high-dimensional optimization problems, and their computation is generally laborious. Furthermore, like the previously mentioned algorithms in this category, they can find only one solution in a single run and thus are essentially unsuitable for MOPs, in which a set of Pareto-optimal solutions, rather than a single solution, is desired. Further, these differential-based algorithms are not applicable for solving partially differentiable MOPs.

The other category consists of population-based heuristic optimization algorithms, e.g., particle swarm optimization [14], [15]; evolutionary algorithms [16]–[24]; and ant colony optimization [25], [26]. Unlike classical differential-based optimization algorithms, these heuristic algorithms have no assumptions regarding the objective functions, such as modality or differentiability, and can find a set of optimal solutions in a single run. Hence, they are much more appropriate for nondifferentiable multimodal optimization problems and MOPs. These algorithms are good at exploring and exploiting promising regions of the search space. However, they suffer from the curse-of-dimensionality problem, as described in [27], and the convergence speed and searching efficiency of these algorithms are both lower than those of classical optimization algorithms. In view of this, some local search strategies [28] have been introduced as separate processes for accelerating the search speed of heuristic algorithms [14], [29]–[31]. Recent studies have demonstrated that these strategies enable more efficient convergence to high-quality solutions on many real-world applications [26], [32], [33]. Nevertheless, these algorithms treat the problem as a "black" box. Thus, these heuristic algorithms still do not take full advantage of the problem properties. Evidently, it is still desirable to improve the search process of an algorithm by considering the nature of the problem of interest.

As far as we know, partially differentiable optimization problems have yet to be well studied in the literature. Therefore, we address such problems in this paper. We propose two decomposition schemes, i.e., objective decomposition and domain decomposition, for solving such problems. In the objective decomposition scheme, the original objective function, which may be either a single- or multiobjective function, is decomposed to obtain an MOP with a greater number of relatively simple subobjectives. We alternately optimize each subobjective by using the Tchebycheff approach [34]. In each procedure, we optimize only one subobjective. Thus, we can isolate the nondifferentiable part of the objective function. In the domain decomposition scheme, the domain of the nondifferentiable subobjective is decomposed into a differentiable subdomain and a nondifferentiable subdomain to isolate the nondifferentiable domain of the subobjective.

In this paper, we propose an effective optimization algorithm, called the simulated water-stream algorithm (SWA), for solving MOPs. A preliminary version of this paper was presented in [35]. In this paper, we introduce a domain decomposition strategy for isolating the nondifferentiable domain of a nondifferentiable subobjective and analyze the convergence of the SWA. The proposed SWA is inspired by the natural phenomenon of water streams moving toward a basin. This process is analogous to the process of finding the minimal solutions for an optimization problem. Water streams generally exhibit two forms of movement: downstream and penetration. Specifically, the process of downstream movement is analogous to a deterministic search, which can make the solution rapidly converge to a stagnation point. If the objective function is differentiable in the subdomain, we directly formulate the deterministic search direction as the gradient of the objective function; otherwise, we approximate the objective function with a kernel density estimator and then formulate the search direction as the gradient of the density estimator. The process of penetration is characterized by two features: 1) necessity and 2) contingency. Necessity means that most of the water streams will penetrate to the lowest location found by their neighboring streams. This is represented by an adaptive cooperative learning heuristic search, in which the search step is adjusted in accordance with the speed of the downstream movement. Contingency refers to the fact that a small portion of the water streams may penetrate laterally or even upward. A random perturbation heuristic search is introduced to simulate the contingency of water penetration. As a result, the SWA combines deterministic and heuristic search approaches in a single framework. It incorporates the advantages of both the methods through the two processes described above. Empirical results fully demonstrate the effectiveness and competitiveness of the proposed algorithm in comparison with its existing counterparts.

The remainder of this paper is organized as follows. Section II describes the application of the objective and domain decomposition strategies to optimization problems. The proposed SWA is elaborated in Section III, and its global convergence is shown in Section IV. We compare the SWA with its existing counterparts in Section V. Finally, we draw the conclusions in Section VI.

II. Alternate Optimization of Each Subobjective Based on Objective and Domain Decomposition

A. *Objective Decomposition*

In many practical applications, the objective of problem (1) can be formulated as the sum of a differentiable function and a nondifferentiable function, that is,

$$F_i(\mathbf{x}) = f_{i,1}(\mathbf{x}) + f_{i,2}(\mathbf{x}) \tag{2}$$

where $f_{i,1}(\mathbf{x})$ is a differentiable function and $f_{i,2}(\mathbf{x})$ is a nondifferentiable one. Therefore, $F_i(\mathbf{x})$ is nondifferentiable as a whole, but part of which is differentiable. An example is the sparse regularization used in many experimental studies. Such an objective function can be decomposed into one differentiable function and one nondifferentiable function. Evidently, when $F_i(\mathbf{x})$ cannot be formulated as a sum of two functions, $f_{i,1}(\mathbf{x})$ is null if $F_i(\mathbf{x})$ is nondifferentiable; otherwise, $f_{i,2}(\mathbf{x})$ is null as $F_i(\mathbf{x})$ is differentiable.

We then merge all of the objective functions decomposed in this way to constitute the following MOP:

$$\min_{\mathbf{x}\in D} f(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^{\mathrm{T}} \tag{3}$$

where $m \leq 2k$ is the number of objective functions of the decomposed problem as expressed in (3). A solution $\mathbf{x}$ is said to Pareto dominate $\mathbf{y}$, denoted by $\mathbf{x} \preceq \mathbf{y}$, if and only if $f_i(\mathbf{x}) \leq f_i(\mathbf{y})\ \forall\ i\ (i = 1, \ldots, m)$ and there exists an i_0 such that $f_{i_0}(\mathbf{x}) < f_{i_0}(\mathbf{y})$. A solution $\bar{\mathbf{x}} \in D$ is a Pareto-optimal solution if $\nexists\ \mathbf{x} \in D$ such that $\mathbf{x} \preceq \bar{\mathbf{x}}$. The set of all Pareto-optimal solutions, called the *Pareto set* (PS), in D is denoted by $E(f, D) \subset D$, and the set of all Pareto-optimal objective vectors is the Pareto front (PF). The optimal solution to problem (1) must be a Pareto-optimal solution to problem (3), as described in Theorem 1.

Theorem 1: $\forall\ \bar{\mathbf{x}} \in E(F, D)$ for problem (1), we have that $\bar{\mathbf{x}} \in E(f, D)$ for problem (3), i.e., $E(F, D) \subseteq E(f, D)$.

Proof: We shall prove $\bar{\mathbf{x}}$ to be a Pareto-optimal solution to problem (3) when $\bar{\mathbf{x}} \in E(F, D)$. Suppose that $\bar{\mathbf{x}} \notin E(f, D)$. According to the definition of a Pareto-optimal solution, there exists a solution $\mathbf{z} \in D$ such that $f_j(\mathbf{z}) \leq f_j(\bar{\mathbf{x}})\ \forall\ j\ (j = 1, \ldots, m)$ and $f_{j_0}(\mathbf{z}) < f_{j_0}(\bar{\mathbf{x}})$ for some j_0. We have that

$$\begin{aligned} F_i(\mathbf{z}) &= f_{i,1}(\mathbf{z}) + f_{i,2}(\mathbf{z}) \\ &\leq f_{i,1}(\bar{\mathbf{x}}) + f_{i,2}(\bar{\mathbf{x}}) = F_i(\bar{\mathbf{x}}) \end{aligned} \tag{4}$$

for $i = 1, \ldots, k$. Obviously, there is at least one i_0 such that $F_{i_0}(\mathbf{z}) < F_{i_0}(\bar{\mathbf{x}})$, that is, $\bar{\mathbf{x}} \notin E(F, D)$, which contradicts the assumption. This implies that the assumption is wrong. Hence, we know that $\bar{\mathbf{x}}$ is a Pareto-optimal solution to (3) because $\bar{\mathbf{x}} \in E(F, D)$. This means that $E(F, D) \subseteq E(f, D)$. ■

Therefore, for the partial differentiable optimization problems, we can first obtain the extended solution set $E(f, D)$ by solving a relatively simple MOP and then search for the final solution $E(F, D)$. Furthermore, we have the following Theorem 2, as presented in [34].

Theorem 2: Let each $h_i(t)$ $(i = 1, \ldots, m)$ be a monotonically increasing function. We consider the following MOP:

$$\min_{\mathbf{x} \in D}\ h(\mathbf{x}) = (h_1(f_1(\mathbf{x})), \ldots, h_m(f_m(\mathbf{x})))^{\mathrm{T}}. \tag{5}$$

Its Pareto-optimal solution set $E(h, D)$ satisfies $E(f, D) = E(h, D)$.

Proof: We first prove that $E(f, D) \subseteq E(h, D)$. For any $\bar{\mathbf{x}} \in E(f, D)$, suppose that $\bar{\mathbf{x}} \notin E(h, D)$, there exists a $\mathbf{z} \in D$ such that $h_j(f_j(\mathbf{z})) \leq h_j(f_j(\bar{\mathbf{x}}))\ \forall\ j\ (j = 1, \ldots, m)$ and $h_{j_0}(f_{j_0}(\mathbf{z})) < h_{j_0}(f_{j_0}(\bar{\mathbf{x}}))$ for some j_0. Since $h_i(t)$ is monotonically increasing, we have $f_j(\mathbf{z}) \leq f_j(\bar{\mathbf{x}})\ \forall\ j\ (j = 1, \ldots, m)$ and $f_{j_0}(\mathbf{z}) < f_{j_0}(\bar{\mathbf{x}})$ for some j_0, that is, $\mathbf{z} \preceq \bar{\mathbf{x}}$, $\bar{\mathbf{x}} \notin E(f, D)$, which contradicts the assumption. This means that $E(f, D) \subseteq E(h, D)$. In a similar way, we can show that $E(h, D) \subseteq E(f, D)$. Thus, problem (5) has the same Pareto-optimal solutions as problem (3), that is, $E(f, D) = E(h, D)$. ■

According to Theorem 1, for nondifferentiable SOPs and MOPs in which some subobjectives can be formulated as a sum of a differentiable function and a nondifferentiable function, such a problem can be decomposed into a differentiable or partially differentiable MOP with more objective functions. Then, we can further simplify such a decomposed optimization problem with monotonically increasing functions, according to Theorem 2. Hence, we can first solve a relatively simple MOP to obtain an approximation $E(h, D)$ and then search for the final Pareto-optimal solution $\bar{\mathbf{x}}$ to problem (1) in this approximation $E(h, D)$.

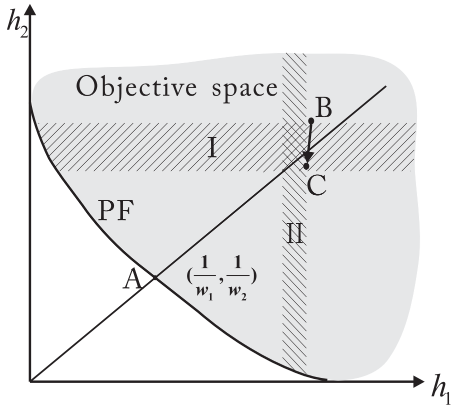


Fig. 1. Illustration of the Tchebycheff approach; the coordinates of points A, B, and C are $(h_1(\mathbf{x}_A), h_2(\mathbf{x}_A))$, $(h_1(\mathbf{x}_0), h_2(\mathbf{x}_0))$, and $(h_1(\mathbf{x}_1), h_2(\mathbf{x}_1))$, respectively.

B. Alternate Optimization of Each Subobjective Based on the Tchebycheff Approach

For simplicity, we assume, in this paper, that all of the objective functions $h_1, \ldots, h_m$ described in (5) are non-negative. Otherwise, we can replace h_i with $h_i - M$, where M is a positive constant. Therefore, all of the objective vectors and the PF of (5) are in R_+^m.

In the Tchebycheff approach [34], problem (5) can be solved by solving the following problem:

$$\min_{\mathbf{x} \in D}\ g(\mathbf{x}|\mathbf{w}) = \min_{\mathbf{x} \in D}\ \max_{1 \leq i \leq m} \{w_i h_i(\mathbf{x})\} \tag{6}$$

where $\mathbf{w} = (w_1, \ldots, w_m)^{\mathrm{T}}$ is a weight vector and $\mathbf{x}$ is the decision variable vector. For any $\bar{\mathbf{x}} \in E(h, D)$ for problem (5), there exists at least one weight vector $\mathbf{w}$ such that $\bar{\mathbf{x}}$ is the optimal solution to problem (6). Let $\{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$ be a set of uniformly distributed weight vectors; then, we can define N subproblems based on (6). We can obtain an approximation of the PF by solving these subproblems. For example, problem (6) is sketched in Fig. 1 for $m = 2$. The equation for the straight line through the origin point with the direction vector $\{(1/w_1), (1/w_2)\}$ is given by $w_1 h_1 = w_2 h_2$. It intersects with the PF at point A, whose coordinates in the objective space are $(h_1(\mathbf{x}_A), h_2(\mathbf{x}_A))$. Obviously,

$$\min_{\mathbf{x} \in D} g(\mathbf{x}|\mathbf{w}) = \min_{\mathbf{x} \in D} \max_{i=1,2} \{w_i h_i(\mathbf{x}_A)\}. \tag{7}$$

That is, $\mathbf{x}_A$ is a weak Pareto-optimal solution to problem (5). The optimization procedure for problem (6) is to search for a solution close to $\mathbf{x}_A$. Moreover, we can draw the following conclusion: for points above the straight line, a smaller objective value h_2 indicates a better point; thus, we have $g(\mathbf{x}_0|\mathbf{w}) = w_2 h_2(\mathbf{x}_0)$. Otherwise, the smaller h_1 is, the better the point is; thus, $g(\mathbf{x}_1|\mathbf{w}) = w_1 h_1(\mathbf{x}_1)$.

The function $g(\mathbf{x}|\mathbf{w})$, however, is a nonsmooth function. We can alternately optimize each subobjective. Suppose that $\mathbf{x}^t$ is the optimal solution obtained by implementing the tth optimization procedure. Then, the optimization problem for the $(t + 1)$th optimization procedure can be expressed as follows:

$$\begin{aligned}\min_{\mathbf{x}\in D}\ & g_{t+1}(\mathbf{x}|\mathbf{w}) = \min_{\mathbf{x}\in D} w_{I_1} h_{I_1}(\mathbf{x})\\ \text{s.t.}\ & w_{I_1} h_{I_1}(\mathbf{x}) \geq (1-\varepsilon) w_{I_2} h_{I_2}\left(\mathbf{x}^t\right)\end{aligned} \tag{8}$$

where $I_1 = \arg\max_{1\leq i\leq m} w_i h_i(\mathbf{x}^t)$, $I_2 = \arg\max_{1\leq i\leq m, i\neq I_1} w_i h_i(\mathbf{x}^t)$, and ε is a positive constant. I_1 and I_2 are the indices of the weighted objectives with the first and second highest values, respectively.

The search procedure for a problem with $m = 2$ is illustrated in Fig. 1. Suppose that the objective vector of the initial solution $\mathbf{x}_0$ corresponds to point B; we expect to search for the optimal point A for the scalar problem, with the weight vector (w_1, w_2). From (8), we know that the current optimization problem is

$$\begin{aligned}\min_{\mathbf{x}\in D}\ & g_1(\mathbf{x}|\mathbf{w}) = \min_{\mathbf{x}\in D} w_2 h_2(\mathbf{x})\\ \text{s.t.}\ & w_2 h_2(\mathbf{x}) \geq (1-\varepsilon) w_1 h_1(\mathbf{x}_0).\end{aligned} \tag{9}$$

As shown in Fig. 1, region I represents the search space for problem (9) in the objective space. We can obtain a new solution $\mathbf{x}_1$ by solving problem (9). Its objective vector corresponds to point C. Then, we construct a new objective function as defined by (8) at point C to search for a new optimal solution in region II.

The advantage of the alternate subobjective optimization method based on the Tchebycheff approach is that we optimize only one simple subobjective in each subproblem. If it is a convex optimization problem, classical optimization algorithms can be used to solve it. Otherwise, heuristic algorithms can be used. This objective decomposition scheme isolates the nondifferentiable objective functions.

C. Domain Decomposition

Let us consider a situation in which a subobjective is nondifferentiable over the whole domain but differentiable in a subdomain. Accordingly, we decompose the domain D of each subobjective $h_i(\mathbf{x})$ $(i = 1, \ldots, m)$ into two types of subdomains, i.e., the nondifferentiable subdomain D_{i1} and the differentiable subdomain D_{i2}. D_{i1} is defined using the bound constraints as a smaller region that contains all points at which $h_i(\mathbf{x})$ is nondifferentiable, and $D_{i2} = D - D_{i1}$. For example, for $p \leq 1$, the regularization term of (25) is nondifferentiable at any point with $x_i = 0$ for $i = 1, \ldots, n$. Thus, the nondifferentiable subdomain D_{i1} is defined as the region with the bound constraint $|x_i| < \varepsilon$ for $i = 1, \ldots, n$, where ε is a small constant.

Subsequently, we need to consider the three cases below.

1) If $h_i(\mathbf{x})$ is differentiable, then $D_{i1} = \Phi$ and $D_{i2} = D$.
2) If $h_i(\mathbf{x})$ is nondifferentiable over the entire domain or we cannot determine its differentiable subdomain, then $D_{i1} = D$ and $D_{i2} = \Phi$.
3) If $h_i(\mathbf{x})$ is partially nondifferentiable, i.e., we can conveniently determine the nondifferentiable subdomain, then we decompose the domain D into a nondifferentiable subdomain D_{i1} and a differentiable subdomain D_{i2}.

This simple domain decomposition scheme isolates the nondifferentiable subdomain of a nondifferentiable subobjective. Therefore, we can adopt different optimization methods to optimize the subobjective in the different subdomains. It is of great significance to present an algorithm that combines deterministic search and heuristic search in a single framework and optimizes different subobjectives with different schemes according to the objective function properties. To this end, we propose the SWA.

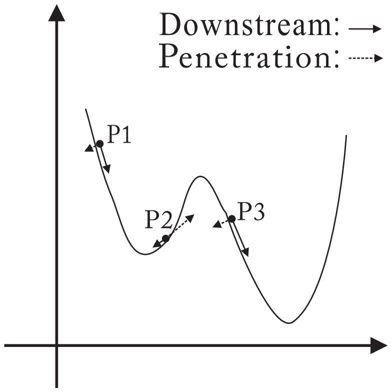


Fig. 2. Solution search procedure of the SWA.

III. Simulated Water-Stream Algorithm

The SWA is a nature-inspired population-based optimization algorithm. It contains N water streams with initial locations of $X^0 = \{\mathbf{x}_1^0, \ldots, \mathbf{x}_N^0\}$. Each water stream flows to a basin via a hybrid process of downstream and penetration. Correspondingly, let $\{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$ be a set of uniformly distributed weight vectors; then, we can define N subproblems based on (8) and optimize each subproblem with a water stream. Because different subproblems have different weight vectors, different subproblems generally yield different solutions. The process of finding the minimum solution can be regarded as a hybrid search process consisting of the downstream movement and penetration of each water stream, as shown in Fig. 2. In the following, we describe the downstream and penetration operators in detail.

A. Downstream Operator

Suppose that the locations of the water streams after the tth fluxion are $X^t = \{\mathbf{x}_1^t, \ldots, \mathbf{x}_N^t\}$. For the ith water stream, the search direction of the downstream operator is the descent direction of the function $g(\mathbf{x}|\mathbf{w}^i)$ at location $\mathbf{x}_i^t$. If $g(\mathbf{x}|\mathbf{w}^i)$ is differentiable at location $\mathbf{x}_i^t$, then the search direction of the downstream operator is given by the gradient of $g(\mathbf{x}|\mathbf{w}^i)$ at $\mathbf{x}_i^t$. Otherwise, we approximate the function $g(\mathbf{x}|\mathbf{w}^i)$ with a kernel density estimator and formulate the search direction as the gradient of the density estimator.

1) Direct Gradient: If the function $g(\mathbf{x}|\mathbf{w}^i)$ is differentiable at location $\mathbf{x}_i^t$, then the search direction of the downstream operator is directly given by the gradient of $g(\mathbf{x}|\mathbf{w}^i)$. Then, the displacement $\mathbf{p}_i^t$ of the water stream can be formulated as follows:

$$\mathbf{p}_i^t = -\alpha_i^t \nabla g\left(\mathbf{x}_i^t|\mathbf{w}^i\right) = -\alpha_i^t w_{I_1} \nabla h_{I_1}\left(\mathbf{x}_i^t\right) \tag{10}$$

where I_1 is defined as in (8) and $\nabla h_{I_1}(\mathbf{x}_i^t)$ is the gradient of $h_{I_1}(\mathbf{x})$ at location $\mathbf{x}_i^t$. The positive scalar α_i^t is called the step length, which can be computed as follows. Since a linear approximation of $g(\mathbf{x}|\mathbf{w}^i)$ at location $\mathbf{x}_i^t$ is

$$g(\mathbf{x}|\mathbf{w}^i) \approx \nabla g(\mathbf{x}_i^t|\mathbf{w}^i)^{\mathrm{T}}\mathbf{x} + b \tag{11}$$

we have

$$\begin{aligned} g(\mathbf{x}_i^t + \mathbf{p}_i^t|\mathbf{w}^i) &\approx \nabla g(\mathbf{x}_i^t|\mathbf{w}^i)^{\mathrm{T}}(\mathbf{x}_i^t + \mathbf{p}_i^t) + b \\ &= \nabla g(\mathbf{x}_i^t|\mathbf{w}^i)^{\mathrm{T}}\mathbf{x}_i^t + b - \alpha_i^t \nabla g(\mathbf{x}_i^t|\mathbf{w}^i)^{\mathrm{T}} \nabla g(\mathbf{x}_i^t|\mathbf{w}^i) \\ &= g(\mathbf{x}_i^t|\mathbf{w}^i) - \alpha_i^t \nabla g(\mathbf{x}_i^t|\mathbf{w}^i)^{\mathrm{T}} \nabla g(\mathbf{x}_i^t|\mathbf{w}^i). \end{aligned}$$

Since $g(\mathbf{x}|\mathbf{w}^i) = g(\mathbf{x}_i^t + \mathbf{p}_i^t|\mathbf{w}^i) \geq (1 - \varepsilon) w_{I_2} h_{I_2}(\mathbf{x}_i^t)$, by substituting $g(\mathbf{x}_i^t + \mathbf{p}_i^t|\mathbf{w}^i)$ into this inequality, we find that $\alpha_i^t \leq ([w_{I_1} h_{I_1}(\mathbf{x}_i^t) - (1-\varepsilon) w_{I_2} h_{I_2}(\mathbf{x}_i^t)]/[\nabla g(\mathbf{x}_i^t|w^i)^{\mathrm{T}} \nabla g(\mathbf{x}_i^t|w^i)])$. Therefore, α_i^t can be computed as follows:

$$\alpha_i^t = \frac{w_{I_1} h_{I_1}(\mathbf{x}_i^t) - (1-\varepsilon) w_{I_2} h_{I_2}(\mathbf{x}_i^t)}{\nabla g(\mathbf{x}_i^t|\mathbf{w}^i)^{\mathrm{T}} \nabla g(\mathbf{x}_i^t|\mathbf{w}^i) + \mathcal{C}} \tag{12}$$

where $\mathcal{C} = 0.1$ is a constant that makes the denominator greater than 0.

2) Approximative Gradient: If the function $g(\mathbf{x}|\mathbf{w}^i)$ is nondifferentiable at location $\mathbf{x}_i^t$, then an approximative gradient can be obtained via nondifferentiable direct search algorithms (e.g., the Hooke–Jeeves algorithm [36] or the Powell algorithm [37]). The displacement $\mathbf{p}_i^t$ of the downstream operator is the approximative gradient. In this paper, we apply a kernel density estimator to approximate the function $g(\mathbf{x}|\mathbf{w}^i)$ and formulate the search direction of the downstream operator as the gradient of the kernel density estimator.

Specifically, we uniformly generate M trial points $Y_M = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_M\}$ in a hypercube centered on the location $\mathbf{x}_i^t$. Since we generally wish to find the maximum value of the density estimator, we define the function

$$\max G(\mathbf{x}) = (\Psi - g(\mathbf{x}|\mathbf{w}^i)) \tag{13}$$

where $\Psi = \max_{1 \leq j \leq M} g(\mathbf{y}_j|\mathbf{w}^i)$. Then, the function $G(\mathbf{x})$ at point $\mathbf{x}$ can be estimated by the following kernel density estimator [38]:

$$\hat{G}_Y(\mathbf{x}) = \frac{1}{\mathbb{C}} \sum_{j=1}^{M} G(\mathbf{y}_j) K\left(\frac{\mathbf{x} - \mathbf{y}_j}{h}\right) \tag{14}$$

where $\mathbb{C}$ is a normalization constant, $K(\mathbf{x})$ is a kernel function, and $h > 0$ is the bandwidth parameter. Then, we obtain the following result.

Proposition 1: When a normal kernel is employed in (14), i.e., $K(\mathbf{x}) = (2\pi)^{-n/2} \exp(-[1/2]\|\mathbf{x}\|^2)$, the search direction

$$\mathbf{p}_i^t = \frac{\sum_{j=1}^{M} G(\mathbf{y}_j) \exp\left(-\frac{1}{2}\left\|\frac{\mathbf{x}_i^t - \mathbf{y}_j}{h}\right\|^2\right)\mathbf{y}_j}{\sum_{j=1}^{M} G(\mathbf{y}_j) \exp\left(-\frac{1}{2}\left\|\frac{\mathbf{x}_i^t - \mathbf{y}_j}{h}\right\|^2\right)} - \mathbf{x}_i^t \tag{15}$$

is the descent direction of $\hat{G}(\mathbf{x})$ at point $\mathbf{x}_i^t$, and it is also the descent direction of $G(\mathbf{x})$ with probability one.

The proof of this proposition is given in [39]. For each trial point $\mathbf{y}_j$, we compute the value of only one subobjective. Hence, the computational complexity does not increase with the number of objectives. Moreover, the displacement of the water stream occurs in the descent direction of the objective function because $\hat{G}(\mathbf{x}_i^t + \mathbf{p}_i^t) > \hat{G}(\mathbf{x}_i^t)$. The details of the procedure for computing the displacement $\mathbf{p}_i^t$ are given in Algorithm 1.

Algorithm 1: Downstream Operation on Water Stream i

```
Input :
  • The location of the water stream $\mathbf{x}_i^t$;
  • The objective function vector $h(\mathbf{x}_i^t)$;
  • The weight vector $\mathbf{w}^i$;
  • The nondifferentiable subdomain $D_{i1}$ and the
    differentiable subdomain $D_{i2}$ of each subobjective,
    $i = 1, 2, \cdots, m$.
Output: The displacement $\mathbf{p}_i^t$.
  Compute the indices $I_1$ and $I_2$ defined in Eq. (8).
if $\mathbf{x}_i^t \in D_{I_1 1}$ then
    Calculate the step length $\alpha_i^t$ from Eq. (12);
    $\mathbf{p}_i^t \leftarrow -\alpha_i^t w_{I_1} \nabla h_{I_1}(\mathbf{x}_i^t)$
else
    Generate $M$ trial points $\{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_M\}$ and
    calculate the values of $G(\mathbf{y}_j)$, $j = 1, \cdots, M$;
    $\mathbf{p}_i^t \leftarrow \frac{\sum_{j=1}^{M} G(\mathbf{y}_j)\exp\left(-\frac{1}{2}\left\|\frac{\mathbf{x}_i^t-\mathbf{y}_j}{h}\right\|^2\right)\mathbf{y}_j}{\sum_{j=1}^{M} G(\mathbf{y}_j)\exp\left(-\frac{1}{2}\left\|\frac{\mathbf{x}_i^t-\mathbf{y}_j}{h}\right\|^2\right)} - \mathbf{x}_i^t$
end
```

The displacement $\mathbf{p}_i^t$ of water stream i obtained via the downstream operator is represented by the arrow with the solid line in Fig. 2. Thus, the new location $\mathbf{x}_i^D$ of the ith water stream can be given as follows:

$$\mathbf{x}_i^D = \mathbf{x}_i^t + \mathbf{p}_i^t, \quad i = 1, \ldots, N. \tag{16}$$

B. Penetration Operator

Water stream penetration has the qualities of both necessity and contingency. In general, most water streams penetrate toward a nearby basin, and the permeability is affected by the downstream movement. The stream flows with higher flow rates will have reduced permeability. Accordingly, a cooperative learning heuristic search approach is used to simulate necessity, in which the search step is self-adjusted according to the downstream speed. In addition, a few water streams may run in any direction, even upward. Thus, a random perturbation is introduced to simulate the contingency of water penetration. The direction of water penetration is indicated by the arrow with the dotted line in Fig. 2. In the following, we describe the cooperative learning heuristic search process and the random perturbation search process in detail.

1) Adaptive Cooperative Learning Heuristic Search: This search process is used to simulate the phenomenon that most water streams penetrate toward the nearby basin. For each water stream $i = 1, \ldots, N$, its neighborhoods are the first K water streams whose weight vectors are closest to $\mathbf{w}^i$, where K is the number of neighbors. The set of indices of the neighborhoods of water stream i is denoted by $B(i)$. Suppose that this search is performed on water stream i; the water stream penetrates toward the lowest nearby location found by one of its neighbors. Thus, the new location of water stream i after penetration is given as follows:

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^D + \lambda_i^t \big(\mathbf{x}_r^L - \mathbf{x}_i^D\big) \tag{17}$$

where λ_i^t is the speed of penetration and r is randomly selected from $B(i)$. The search is adaptively adjusted according to the downstream speed of the water displacement $\mathbf{p}_i^t$, that is

$$\lambda_i^t = 0.1 r_1 \cdot e^{-\|\mathbf{p}_i^t\|} \tag{18}$$

where r_1 is a uniform random number in [0, 1] and $\|\mathbf{p}_i^t\|$ is the norm of $\mathbf{p}_i^t$. Equation (18) implies that the closer to an extreme point $\mathbf{x}_i^t$ is, the higher the speed of penetration is. Thus, the penetration process will dominate the search for points near extreme points. This facilitates escape when a solution is close to a local minimum and avoids the premature convergence of the algorithm. For example, the penetration of P2 is small because it is close to a local minimum, as shown in Fig. 2. However, when $\mathbf{x}_i^t$ is far away from the extreme points, (18) leads to a relatively low speed of penetration. Thus, the deterministic search dominates the search procedure. Positions P1 and P3 in Fig. 2 are consistent with the situation described above. Obviously, this situation is beneficial for the convergence of the algorithm.

2) Random Perturbation Heuristic Search: A random perturbation heuristic search process is introduced to simulate the contingency of water penetration. This search process is performed on only a small portion of the water streams. Suppose that this random perturbation process is performed on the ith water stream. Every component of $\mathbf{x}_i^D$ is perturbed with a given probability p. If $x_{i,l}^D$, the lth component of $\mathbf{x}_i^D$, is selected to be perturbed, it reassigned a uniform random value in $[a_l, b_l]$. Hence, its new location is computed as follows:

$$x_{i,l}^{t+1} = \begin{cases} a_l + r_2(b_l - a_l) & \text{if } r < p \\ x_{i,l}^D & \text{otherwise} \end{cases} \tag{19}$$

with $l = 1, \ldots, n$, where r, like r_1 in (18), is a uniform random number in [0, 1]. This random perturbation process is beneficial for maintaining the diversity of the solutions and avoiding premature convergence. The overall flow of the SWA is summarized in Algorithm 2, where the following are maintained at each fluxion t.

1) The current locations $X^t = \{\mathbf{x}_1^t, \ldots, \mathbf{x}_N^t\}$ and the lowest locations $X^L = \{\mathbf{x}_1^L, \ldots, \mathbf{x}_N^L\}$, where $\mathbf{x}_i^t$ and $\mathbf{x}_i^L$ are the current and lowest locations, respectively, of the ith water stream.
2) The objective values $f(\mathbf{x}_i^L)$, $i = 1, \ldots, N$.
3) An external population (EP), which is used to store optimal solutions to the original problem found during the search.

IV. Analysis of the Global Convergence of the SWA

The MOP in (3) can be transformed into a number of SOPs with the form of (6) by using the Tchebycheff approach with different weight vectors. Therefore, we show only the SWA is a global search algorithm for SOPs.

Algorithm 2: SWA

```
Input :
  • $N$: the water stream size;
  • $K$: the neighborhood size;
  • $p$: the perturbation probability;
  • $MF$: the maximum number of fluxions;
  • The original problem (1);
  • The decomposed problem (5).
Output: EP.
{Initialization:}
Uniformly generate $N$ weight vectors $\{\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^N\}$.
Construct $B(i)$ as the set of indices of the $K$ closest
weight vectors to $\mathbf{w}^i$, and initialize EP$= \emptyset$. Randomly
sample $N$ initial locations $X^0 = \{\mathbf{x}_1^0, \cdots, \mathbf{x}_N^0\}$ from the
decision space, and set $X^L = X^0$.
for $t \leftarrow 1$ to $MF$ do
    for $i \leftarrow 1$ to $N$ do
        {Downstream Operator:}
        Apply the downstream operator to $\mathbf{x}_i^t$ to produce
        the displacement $\mathbf{p}_i^t$ via Algorithm 1 and obtain
        the new location $\mathbf{x}_i^D$ from Eq.(16).
        {Penetration Operator:}
        Let $r_3$ be a random number from rand.
        if $r_3 > 0.1$ then
            Generate the new solution $\mathbf{x}_i^{t+1}$ via Eq.(17).
        else
            Generate the new solution $\mathbf{x}_i^{t+1}$ via Eq.(19).
        end
        {Update the lowest location:}
        foreach $j \in B(i)$ do
            if $g(\mathbf{x}_i^{t+1}|\mathbf{w}^j) < g(\mathbf{x}_j^L|\mathbf{w}^j)$ then
                $\mathbf{x}_j^L = \mathbf{x}_i^{t+1}$ and $h(\mathbf{x}_j^L) = h(\mathbf{x}_i^{t+1})$.
            end
        end
    end
    {Update of EP:}
    Remove from EP all individuals dominated by $F(\mathbf{x}_j^L)$;
    Add $F(\mathbf{x}_j^L)$ to EP if no vectors in EP dominate $F(\mathbf{x}_j^L)$;
    $t = t + 1$.
end
```

A. Overview of the Theoretical Results

Suppose that the global minimizer of the problem in (6) is not isolated, that is, its sublevel set

$$D_\beta = \{\mathbf{x} \in D \mid g(\mathbf{x}|w) \le \beta\} \tag{20}$$

for $\beta > \min_{\mathbf{x} \in D} g(\mathbf{x}|\mathbf{w})$ is a nonempty and compact set. In [40] and [41], Solis and Wets have provided a criterion for determining whether an algorithm is a global search algorithm based on two assumptions and one theorem, as follows.

A1: $g(H(\mathbf{z}, \xi)|\mathbf{w}) \le g(\mathbf{z}|\mathbf{w})$ and if $\xi \in D$, then $g(H(\mathbf{z}, \xi)|\mathbf{w}) \le g(\xi|\mathbf{w})$, where H is a function that constructs a solution to the problem. This assumption guarantees that the newly constructed solution will be no worse than the current one.

A2: For any subset A of D with $v(A) > 0$, we have

$$\prod_{k=0}^{\infty}(1-\mu_k(A)) = 0 \tag{21}$$

where $v(A)$ is the n-dimensional volume of the set A, $\mu_k(A)$ is the probability of A being generated by μ_k, and μ_k is a probability measure. This assumption implies that for any subset A of D with a positive v, the probability of repeatedly missing the set A must be zero.

Theorem 3: Suppose that g is a measurable function and that D is a measurable subset of R^n. If an algorithm satisfies A1 and A2, then it converges to a globally optimal solution with probability one.

B. SWA: Global Search Algorithm

In this section, we utilize the results of the paper [40] to study the convergence characteristics of the SWA. The SWA simulates the downstream movement and penetration of water streams. More precisely, it proceeds through the following processes:

$$X^L \xrightarrow{\text{Down}(X^t)} X^D \xrightarrow{\text{Penetrate}} X^{t+1} \xrightarrow{\text{Update}} X^{'L}$$

where $X^{'L}$ represents the new lowest locations of the water streams.

Lemma 1: The SWA satisfies A1.

Proof: From the solution update in the SWA, we know that the function H (as introduced in A1) is defined as follows:

$$H\big(\delta\big(\mathbf{x}_i^t\big), \mathbf{x}_i^t\big) = \begin{cases} \mathbf{x}_i^t & \text{if } g\big(\delta\big(\mathbf{x}_i^t\big)\big) \geq g\big(\mathbf{x}_i^t\big) \\ \delta\big(\mathbf{x}_i^t\big) & \text{otherwise} \end{cases} \tag{22}$$

where δ denotes the functions corresponding to the operators applied in the SWA as defined in (16), (17), and (19). The above definition of H clearly complies with A1. From Lemma 1, we know that the lowest locations of the water streams are monotonically decreasing and bounded. Thus, the algorithm is convergent. ■

Lemma 2: The SWA satisfies A2.

Proof: The penetration process of the SWA is a heuristic search process. To satisfy A2, the union of the sample spaces of the solutions must cover D, such that

$$D \subseteq \bigcup_{i=1}^{N} M_i^t \tag{23}$$

where M_i^t is the support of the sample space of $\mathbf{x}_i^D$. There are two different definitions for M_i^t.

1) For the solutions updated with (17), whose index set is denoted by Δ_1, the shape of M_i^t is defined as follows:

$$M_i^t = \mathbf{x}_i^D + \lambda_i^t \cdot \Omega_i^t \tag{24}$$

where $\Omega_i^t = \mathbf{x}_i^L - \mathbf{x}_i^D$. M_i^t is a hyper-rectangle parameterized by λ_i^t, with one corner specified by $\lambda_i^t = 0$ and the other by $\lambda_i^t = 0.1e^{-\|\mathbf{p}_i^t\|}$. For $0.1e^{-\|\mathbf{p}_i^t\|} \cdot \Omega_i^t < 0.5\text{diam}(D)$, it is clear that $v(M_i^t \cap D) < v(D)$, where diam(D) denotes the length of D. Since the SWA is convergent, the length of M_i^t will tend toward 0 as t tends toward infinity. Thus, $v(\cup_{i\in\Delta_1} M_i^t \cap D) < v(D)$, i.e., M_i^t cannot cover D for $i \in \Delta_1$.

2) For the solutions updated with (19), whose index set is denoted by Δ_2, it is clear that $M_i^t = D$ for $i \in \Delta_2$.

In summary, we have $D \subseteq \cup_{i\in\Delta_2} M_i^t \subseteq \cup_{i=1}^{N} M_i^t$, which implies that the SWA satisfies A2. ■

Theorem 4: The SWA converges to a globally optimal solution with probability one.

Proof: The SWA satisfies A1 and A2 by Lemmas 1 and 2. According to Theorem 3, the SWA converges to a globally optimal solution with probability one. ■

V. Experimental Simulations

A. Experimental Design

We conducted the following three experiments to evaluate the performance of the proposed SWA.

1) *Practical Application Problems:* We compared the proposed algorithm with two classical iterative algorithms, i.e., the Lasso algorithm [42] and Xu's algorithm [43], on sparse regularization with the ℓ_1-norm and the $\ell_{(1/2)}$-norm, separately. The Lasso algorithm proposed by R. Tibshirani has been widely used for regularization with the ℓ_1-norm. An iterative algorithm for regularization with the $\ell_{(1/2)}$-norm was proposed in [43].
2) *SOPs:* We applied the SWA to SOPs to evaluate its convergence speed and precision. We compared the SWA with the PSwarm[1] [44] and EA [20] approaches on several SOPs.
3) *MOPs:* Currently, various evolutionary multiobjective algorithms have been presented for solving MOPs. We chose MOEA/D and NSGA-II as baselines for our experiments because MOEA/D is one of the most popular decomposition-based EMO algorithms, while NSGA-II is one of the most popular dominance-based EMO algorithms. We compared the proposed SWA with MOEA/D and NSGA-II and effectively verified the performance of the SWA for solving MOPs.

B. Experiment 1

Sparse regularization is often used in practical applications for variable selection and feature extraction. Given $A \in \mathbf{R}^{l\times n}$ and $Y \in \mathbf{R}^l$ as the vector of observations, the regularization task can be formulated as follows:

$$\min_{\mathbf{x}} F(\mathbf{x}) = \|Y - A\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_p \tag{25}$$

where $\lambda > 0$ is the regularization coefficient and $\mathbf{x} \in \mathbf{R}^n$ is a vector of unknowns. The regularization output is the sum of the regression error $\|Y - A\mathbf{x}\|_2^2$ and the regularization term $\|\mathbf{x}\|_p$. $\|\mathbf{x}\|_p$ is the ℓ_p-norm of $\mathbf{x}$, where $p \geq 0$. When $p \leq 1$, regularization is known to produce sparse coefficients and may identify irrelevant features. The regularization term is a nondifferentiable nonconvex function when $p \leq 1$. Therefore, the sparse regularization problem cannot be directly solved by using differential-based optimization algorithms. Problem (25) has an objective function that is a sum of two functions and can be decomposed into the following MOP:

$$\begin{cases} f_1(\mathbf{x}) = \|Y - A\mathbf{x}\|_2^2 \\ f_2(\mathbf{x}) = \|\mathbf{x}\|_p. \end{cases} \tag{26}$$

This MOP can then solved by using the proposed SWA. According to Theorem 1, the optimal solution to the original

[1] http://www.norg.uminho.pt.

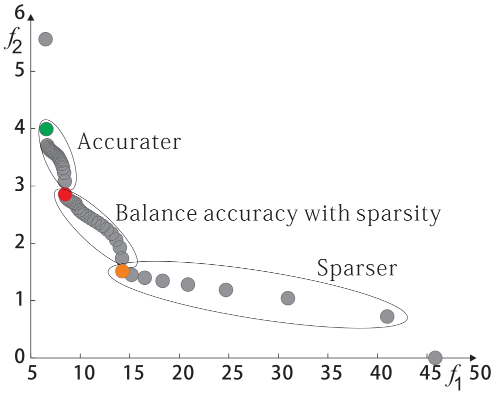


Fig. 3. Final population in the objective space of problem (26) with $p = 0.5$ obtained by the SWA on the first dataset.

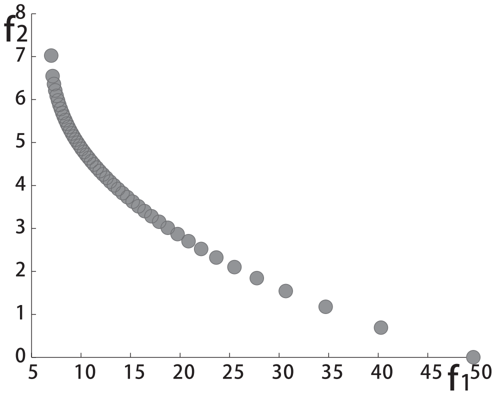


Fig. 4. Final population in the objective space of problem (26) with $p = 1$ obtained by the SWA on the first dataset.

problem given in (25) must be a Pareto-optimal solution to the decomposed problem given in (26). We note that we need not consider the regularization coefficient λ in the decomposed problem.

We present the performance evaluations conducted using variable selection as our example application [1], [42]. In this paper, 100 datasets are considered. Each dataset consists of 100 observations sampled from the following linear model:

$$Y = A\beta + \sigma\varepsilon$$

where $A = (a_1, \ldots, a_8)$, $\beta = (3, 1.5, 0, 0, 2, 0, 0, 0)^{\mathrm{T}}$, $\sigma = 3$, and ε is a random error drawn from the standard normal distribution plus 30% outliers from the standard Cauchy distribution. a_i $(i = 1, \ldots, 8)$ obeys a normal distribution, and the correlation between a_i and a_j is $\rho^{|i-j|}$, with $\rho = 0.5$. We assume that $-1 \leq x_i \leq 5$ for $i = 1, \ldots, 8$ because $\mathbf{x}$ is an estimate of β.

For Experiment 1, the parameters of the proposed SWA were set as follows.

1) The water stream size is $N = 50$.
2) The neighborhood size for each water stream is $K = 5$.
3) The perturbation probability is $p = 0.1$.
4) $\mathrm{MF} = 50$ and $\mathrm{MF} = 30$ for regularization with the ℓ_1-norm and the $\ell_{(1/2)}$-norm, respectively.

When the absolute value of an element of $\mathbf{x}$ is smaller than 0.001, we consider it to be zero. We measure the sparsity of $\mathbf{x}$ by the number of zero elements of $\mathbf{x}$, denoted by $\mathrm{Deg}(\mathbf{x})$. Fig. 3 shows the distribution of the final solutions found by the SWA for the optimization problem given in (26) with the $\ell_{(1/2)}$-norm on the first dataset. It can be observed that the PF is piecewise concave. The left part of the PF, with $\mathrm{Deg}(\mathbf{x}) = 3$, offers a high reconstruction accuracy. The middle part, with $\mathrm{Deg}(\mathbf{x}) = 2$, provides a balance between the accuracy and sparsity. The Pareto-optimal solution corresponding to the right part, with $\mathrm{Deg}(\mathbf{x}) = 1$, is sparse for the problem. Fig. 4 plots the final solution found by the SWA for the optimization problem given in (26) with the ℓ_1-norm on the first dataset. There is no obvious border between Pareto-optimal solutions with different sparsities.

The average accuracy, which is defined as the average number of correctly identified zero elements (CAN) over the 100 tests, is used to measure the performance. If all zero elements are correctly identified, the CAN value is 5 in this experiment. That is, the maximum CAN value is 5. In addition, a set of solutions is obtained by the SWA in a single run. Table I lists the proportions of the solutions with different sparsities in the population, except for minority solutions with $\mathrm{Deg}(\mathbf{x}) > 3$. Table I also shows the CAN values achieved by Xu's algorithm and the SWA for regularization with the $\ell_{(1/2)}$-norm. It can be observed that the CAN values for the majority of the solutions obtained by the SWA are higher than the CAN for Xu's algorithm. This means that the SWA has a higher accuracy than Xu's algorithm. Table II shows the results obtained by the Lasso algorithm and the SWA for regularization with the ℓ_1-norm. As shown in this table, the CAN values for the solutions with $\mathrm{Deg}(\mathbf{x}) = 2$ and 3 obtained by the SWA are greater than the CAN for the Lasso algorithm. These results show that the SWA can more accurately identify the zero elements.

TABLE I
RESULTS OBTAINED BY XU'S ALGORITHM AND THE SWA FOR $\ell_{(1/2)}$-NORM REGULARIZATION

Algorithm	Deg	Proportion	CAN
Xu's Algorithm	—	—	4.6
SWA	3	0.34	4.81
	2	0.31	4.92
	1	0.25	4.90

TABLE II
RESULTS OBTAINED BY THE LASSO ALGORITHM AND THE SWA FOR ℓ_1-NORM REGULARIZATION

Algorithm	Deg	Proportion	CAN
Lasso	—	—	4.01
SWA	5	0.21	2.78
	4	0.24	3.63
	3	0.15	4.52
	2	0.09	4.79

C. Experiment 2

Rastrigin's function and Ackley's function [20], each of which is formulated as a sum of two functions, are two widely used single-objective optimization test benchmarks. Therefore, the following four test instances, that is, Rastrigin's function (SF1), Ackley's function (SF2), the rotated Rastrigin's function (SF3), and the rotated Ackley's function (SF4), were

TABLE III

BEST AND MEAN OBJECTIVE FUNCTION VALUES FOR THE BEST-SO-FAR SOLUTIONS OBTAINED BY THE SWA, EA, AND PSWARM METHODS OVER 20 INDEPENDENT RUNS

Instance	n	SWA			EA			PSwarm		
		num_fun	best	mean	num_fun	best	mean	num_fun	best	mean
	10	**3,000**	**0**	**0**	12,000	3.481E-12	8.921E-11	10,003	1.761E-8	3.345
SF1	50	**5,000**	**0**	**0**	100,000	8.320E-10	9.221E-7	56,204	1.879E-7	68.98
	100	**10,000**	**0**	**0**	300,000	4.478E-4	1.235	156,956	3.994E-7	53.23
	10	**3,000**	**0**	**0**	12,000	3.429E-10	4.527E-9	17,469	6.964	11.939
SF2	50	**5,000**	**0**	**0**	100,000	2.453E-5	8.378E-4	31,480	68.652	89.546
	100	**10,000**	**0**	**0**	300,000	4.213E-3	6.362E-2	108,856	133.324	203.966
	10	**3,000**	**0**	**0**	12,000	5.563E-8	5.216E-6	12,362	3.479E-8	5.701
SF3	50	**5,000**	**0**	**0**	100,000	2.367E-5	4.763E-3	73,801	4.890	89.470
	100	**10,000**	**0**	**0**	300,000	0.357	2.317	204,812	25.245	128.151
	10	**3,000**	**0**	**0**	12,000	6.472E-10	7.582E-8	20,901	8.2145	21.314
SF4	50	**5,000**	**0**	**0**	100,000	7.271E-7	6.317E-5	85,120	21.891	71.403
	100	**10,000**	**0**	**0**	300,000	2.468E-3	0.368	209,791	150.298	231.192

used as the benchmarks in this experiment to evaluate the performance of the proposed SWA.

SF1 (Rastrigin's Function):

$$F(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n} [10 - 10\cos(2\pi x_i)]$$

where $-5.12 \le x_i \le 5.12$, $i = 1, \dots, n$. Its global minimum value is 0 at $\mathbf{x} = (0, \dots, 0)$. It is a multimodal function and has 10^n local minima.

SF2 (Ackley's Function):

$$F(\mathbf{x}) = -20\exp\left(-\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$$

where $-32 \le x_i \le 32$, $i = 1, \dots, n$. It has 64^n locally optimal solutions in the decision space. Its global minimum value is also 0 at $\mathbf{x} = (0, \dots, 0)$.

In SF1 and SF2, the decision variables are separable and can be solved for with n searches. Two rotated multimodal problems, i.e., the rotated Rastrigin's function and the rotated Ackley's function, are used to test the ability of the proposed algorithm to solve problems with variable coupling.

SF3 (Rotated Rastrigin's Function):

$$F(\mathbf{x}) = \sum_{i=1}^{n} y_i^2 + \sum_{i=1}^{n} \big[3 - 3\cos(2\pi y_i)\big].$$

SF4 (Rotated Ackley's Function):

$$F(\mathbf{x}) = -20\exp\left(-\sqrt{\frac{1}{n}\sum_{i=1}^{n} y_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi y_i)\right) + 20 + e.$$

where, $y_i = \sum_{j=1}^{n} m_{ij}x_j$, $i = 1, \dots, n$, and $M = (m_{ij})_{n\times n}$ is an orthogonal matrix. The rotated functions cannot be solved with only n searches because all dimensions of $\mathbf{y}$ will be affected when one dimension of $\mathbf{x}$ is changed. The optimal solutions to these functions are not affected by the orthogonal rotation matrix. Thus, we have the other two corresponding test instances.

These SOPs can be transformed into the following MOPs via the proposed objective decomposition strategy:

$$\textbf{SF1':}\ \begin{cases} f_1(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 \\ f_2(\mathbf{x}) = \sum_{i=1}^{n} 10 - 10\cos(2\pi x_i) \end{cases}$$

$$\textbf{SF2':}\ \begin{cases} f_1(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i^2 \\ f_2(\mathbf{x}) = \sum_{i=1}^{n} 1 - \cos(2\pi x_i) \end{cases}$$

$$\textbf{SF3':}\ \begin{cases} f_1(\mathbf{x}) = \sum_{i=1}^{n} y_i^2 \\ f_2(\mathbf{x}) = \sum_{i=1}^{n} 3 - 3\cos(2\pi y_i) \end{cases}$$

where $y_i = \sum_{j=1}^{n} m_{ij}x_j$, $i = 1, \dots, n$.

$$\textbf{SF4':}\ \begin{cases} f_1(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} y_i^2 \\ f_2(\mathbf{x}) = \sum_{i=1}^{n} 1 - \cos(2\pi y_i) \end{cases}$$

where $y_i = \sum_{j=1}^{n} m_{ij}x_j$, $i = 1, \dots, n$.

Note that the optimal solutions to SF1–SF4 must be Pareto-optimal solutions to the corresponding decomposed MOPs SF1'–SF4' according to Theorems 1 and 2. After this transformation, the objectives are simplified; in particular, the objectives of SF2 and SF4 are nondifferentiable, but each objective of SF2' and SF4' is differentiable. The SWA was independently executed 20 times on each transformed MOP. The parameters of the SWA were the same as those used in Experiment 1 except that the maximum number of fluxions was set to MF = num_fun/N, where num_fun is the number of function evaluations listed in Table III.

We compared the proposed SWA with two popular heuristic algorithms, i.e., the EA proposed in [20] and PSwarm [44]. Both the EA and PSwarm were independently run 20 times on each of the original SOPs SF1–SF4. Each of them was stopped when a maximum number of function evaluations, also listed in Table I, was reached in each run. For the EA, the population size was set to 100, and the other control parameters for the crossover and mutation operators were the same as those used in [20]. For PSwarm, the particle swarm size was set to 40, and the other parameters were set to their default values. We used the code downloaded from the website: http://www.norg.uminho.pt, to solve these test instances.

TABLE IV
MINIMUM (best) AND AVERAGE (mean) VALUES OF THE IGD METRIC ACHIEVED BY THE SWA, NSGA-II AND MOEA/D IN 20 INDEPENDENT RUNS FOR EACH TEST INSTANCE

Instance	n	SWA			NSGA-II			MOEA/D		
		num_fun	best	mean	num_fun	best	mean	num_fun	best	mean
MF1	10	**10,000**	**0.0025**	**0.0025**	50,000	0.0045	0.0068	50,000	0.0030	0.0035
	50	**30,000**	**0.0031**	**0.0033**	300,000	0.0052	0.0073	300,000	0.0038	0.0053
MF2	10	**10,000**	**0.0019**	**0.0020**	50,000	0.0036	0.0047	50,000	0.0020	0.0021
	50	**30,000**	**0.0020**	**0.0020**	300,000	0.0038	0.0050	300,000	0.0020	0.0023
MF3	10	**10,000**	**0.0028**	**0.0035**	50,000	0.0030	0.0039	50,000	0.0031	0.0035
	50	**30,000**	**0.0030**	**0.0038**	300,000	0.0041	0.0058	300,000	0.0034	0.0068
MF4	10	**30,000**	**0.0235**	**0.0257**	50,000	0.0325	0.0429	50,000	0.0298	0.0328
	50	**50,000**	**0.0248**	**0.0263**	300,000	0.0498	0.0501	300,000	0.0279	0.0319
MF5	10	**30,000**	**0.0423**	**0.0487**	50,000	0.0523	0.0581	50,000	0.0429	0.0507
	50	**50,000**	**0.0512**	**0.0584**	300,000	0.0529	0.0698	300,000	0.0460	0.0529

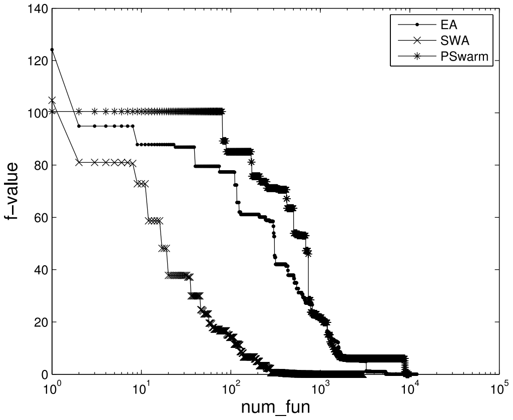


Fig. 5. Objective function values of the best-so-far solutions obtained by the SWA, EA, and PSwarm methods versus num_fun for SF1 with $n = 10$.

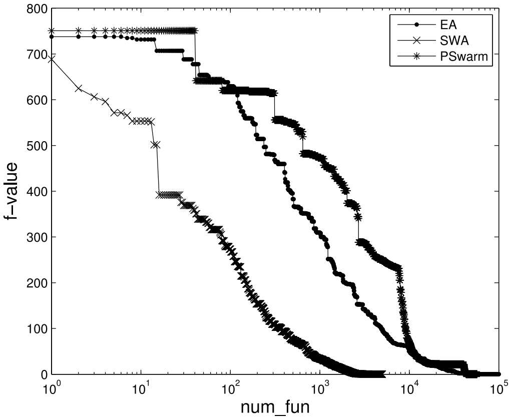


Fig. 6. Objective function values of the best-so-far solutions obtained by the SWA, EA, and PSwarm methods versus num_fun for SF1 with $n = 50$.

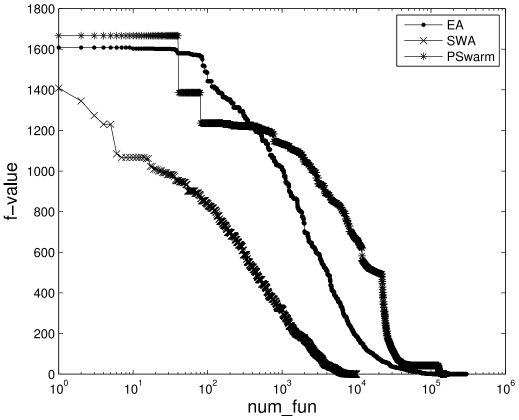


Fig. 7. Objective function values of the best-so-far solutions obtained by the SWA, EA, and PSwarm methods versus num_fun for SF1 with $n = 100$.

The results are compared in Table III in terms of the best and mean objective function values for the best-so-far solutions over 20 independent runs. The results obtained by the SWA are reported as the objective function values for the best-so-far solutions according to the original objective function. It can be seen from Table III that the SWA performs better than the EA and PSwarm for all test instances considered thus far. The SWA can find the optimal solution with fewer function evaluations. By contrast, the EA can find close-to-optimal solutions with more generations, and PSwarm may converge to a locally optimal solution.

Figs. 5–7 plot the objective values of the best-so-far solutions obtained for SF1 by the three methods versus num_fun with $n = 10$, 50, and 100, respectively. From these figures, we can see that the convergence speed of the SWA is faster than those of the EA and PSwarm.

D. Experiment 3

In Experiment 3, we compared the SWA with NSGA-II [45] and MOEA/D [17] on multimodal MOPs. In NSGA-II and MOEA/D, crossover and mutation operators with the same control parameters as the ones used in [17] were used to generate new solutions. The other control parameters in MOEA/D, i.e., $T = 20$ and $\delta = 0.9$, were also the same as those used in [17]. In both experiments, the neighborhood size in the SWA was set to 5, while the perturbation probability was set to 0.1.

In this experiment, five differentiable multiobjective test instances, denoted by MF1–MF5, were constructed based on DTLZ [17]

$$\mathrm{MF1}:\ \begin{cases} f_1(\mathbf{x}) = g(\mathbf{x}) + 1 - x_1 \\ f_2(\mathbf{x}) = g(\mathbf{x}) + x_1 \end{cases}$$

where $g(\mathbf{x}) = \sum_{i=2}^{n}(x_i^2 - 3\cos(10\pi x_i) + 3)$ and $x \in [0, 1] \times [-1, 1]^{n-1}$

$$\text{MF2}: \begin{cases} f_1(\mathbf{x}) = g(\mathbf{x}) + \cos(0.5\pi x_1) \\ f_2(\mathbf{x}) = g(\mathbf{x}) + \sin(0.5\pi x_1) \end{cases}$$

where $g(\mathbf{x})$ and the decision space are the same as in MF1

$$\text{MF3}: \begin{cases} f_1(\mathbf{x}) = g(\mathbf{x}) + 1 - \cos x_1 \\ f_2(\mathbf{x}) = g(\mathbf{x}) + 1 - \sin x_1 \end{cases}$$

where

$$g(\mathbf{x}) = -20\exp\left(-\sqrt{1 + \frac{1}{n}\sum_{i=2}^{n} 10x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=2}^{n}\cos(20\pi x_i)\right) + 20e^{-1} + e$$

and $\mathbf{x} \in [0, 1] \times [-1, 1]^{n-1}$

$$\text{MF4}: \begin{cases} f_1(\mathbf{x}) = g(\mathbf{x}) + \cos(0.5\pi x_1)\cos(0.5\pi x_2) \\ f_2(\mathbf{x}) = g(\mathbf{x}) + \cos(0.5\pi x_1)\sin(0.5\pi x_2) \\ f_3(\mathbf{x}) = g(\mathbf{x}) + \sin(0.5\pi x_1) \end{cases}$$

where $g(\mathbf{x}) = \sum_{i=3}^{n}(x_i^2 - 3\cos(10\pi x_i) + 3)$ and $x \in [0, 1]^2 \times [-1, 1]^{n-2}$

$$\text{MF5}: \begin{cases} f_1(\mathbf{x}) = g(\mathbf{x}) + (1 - \cos(0.5\pi x_1))(1 - \cos(0.5\pi x_2)) \\ f_2(\mathbf{x}) = g(\mathbf{x}) + (1 - \cos(0.5\pi x_1))(1 - \sin(0.5\pi x_2)) \\ f_3(\mathbf{x}) = g(\mathbf{x}) + (1 - \sin(0.5\pi x_1)) \end{cases}$$

where

$$g(\mathbf{x}) = -20\exp\left(-\sqrt{1 + \frac{1}{n}\sum_{i=3}^{n} 10x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=3}^{n}\cos(20\pi x_i)\right) + 20e^{-1} + e$$

and $\mathbf{x} \in [0, 1]^2 \times [-1, 1]^{n-2}$. For MF1–MF3, the population size N was set to 100, while $N = 300$ for MF4–MF5.

The IGD metric is used to evaluate the performance of the algorithms in this paper. Let Q^* be a set of points uniformly distributed along the PF, and let Q be an approximation to the PF obtained via a given algorithm. The IGD metric is defined as the distance between Q^* and Q

$$\text{IGD}\big(Q^*, Q\big) = \frac{\sum_{v \in Q^*} d(v, Q)}{|Q^*|}$$

where $d(v, Q)$ is the minimum Euclidean distance from point v to Q. Obviously, a smaller IGD value indicates better algorithm performance. We uniformly selected 500 points for the 2-objective test instances and 1000 points for the 3-objective test instances to construct the set Q^* along the PF.

For all algorithms, 20 independent runs were executed for each test instance. Table IV presents the best and mean IGD values of the final solutions obtained by each algorithm for each test instance. It can be seen that the SWA outperforms the other algorithms on all test instances considered thus far.

VI. Conclusion

This paper has addressed the optimization problems with partially differentiable objective functions. We have proposed two decomposition schemes: 1) objective decomposition and 2) domain decomposition. In the former, the original objective function is decomposed into several relatively simple subobjectives to isolate the differentiable part of the objective function, thereby formulating the original problem as an MOP. In the latter scheme, the domain is decomposed into two subdomains, i.e., the differentiable and nondifferentiable domains, to isolate the nondifferentiable domain of a nondifferentiable subobjective. Subsequently, we have proposed the novel SWA, which combines deterministic search and heuristic search in a single paradigm, resulting in fast convergence while converging to a globally optimal solution with probability one. Numerical simulations have shown that the SWA yields promising results in comparison with its existing counterparts.

References

[1] H. Zeng and Y.-M. Cheung, “Feature selection and kernel learning for local learning-based clustering,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1532–1547, Aug. 2011.

[2] B. Xue, M. Zhang, W. N. Browne, and X. Yao, “A survey on evolutionary computation approaches to feature selection,” *IEEE Trans. Evol. Comput.*, vol. 20, no. 4, pp. 606–626, Aug. 2016.

[3] L. Zhang *et al.*, “Kernel sparse representation-based classifier,” *IEEE Trans. Signal Process.*, vol. 60, no. 4, pp. 1684–1695, Apr. 2012.

[4] S. Gao, I. W.-H. Tsang, and L.-T. Chia, “Laplacian sparse coding, hypergraph Laplacian sparse coding, and applications,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 92–104, Jan. 2013.

[5] E. Hosseini-Asl, J. M. Zurada, and O. Nasraoui, “Deep learning of part-based representation of data using sparse autoencoders with nonnegativity constraints,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 12, pp. 2486–2498, Dec. 2016.

[6] M. R. Hestenes and E. Stiefel, “Methods of conjugate gradients for solving linear systems,” *J. Res. Nat. Bureau Stand.*, vol. 49, no. 6, pp. 409–436, 1952.

[7] W. C. Davidon, “Variable metric method for minimization,” *SIAM J. Optim.*, vol. 1, no. 1, pp. 1–17, 1991.

[8] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge Univ. Press, 2004.

[9] P. R. Chowdhury, Y. P. Singh, and R. A. Chansarkar, “Hybridization of gradient descent algorithms with dynamic tunneling methods for global optimization,” *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 30, no. 3, pp. 384–390, May 2000.

[10] N. Kazazakis and C. S. Adjiman, “Arbitrarily tight α BB underestimators of general non-linear functions over sub-optimal domains,” *J. Glob. Optim.*, vol. 71, no. 4, pp. 815–844, 2018.

[11] Z. Zhou and F. Bai, “An adaptive framework for costly black-box global optimization based on radial basis function interpolation,” *J. Glob. Optim.*, vol. 70, no. 4, pp. 757–781, 2018.

[12] Q. Han and J. Han, “Revised filled function methods for global optimization,” *Appl. Math. Comput.*, vol. 119, nos. 2–3, pp. 217–228, 2001.

[13] Z. Y. Wu, F. S. Bai, H. W. J. Lee, and Y. J. Yang, “A filled function method for constrained global optimization,” *J. Glob. Optim.*, vol. 39, no. 4, pp. 495–507, 2007.

[14] Y.-J. Gong *et al.*, “Genetic learning particle swarm optimization,” *IEEE Trans. Cybern.*, vol. 46, no. 10, pp. 2277–2290, Oct. 2016.

[15] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga, “Handling multiple objectives with particle swarm optimization,” *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 256–279, Jun. 2004.

[16] K. Deb, *Multiobjective Optimization Using Evolutionary Algorithms*. New York, NJ, USA: Wiley, 2001.

[17] Q. Zhang and H. Li, “MOEA/D: A multiobjective evolutionary algorithm based on decomposition,” *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[18] H.-L. Liu, F. Gu, and Q. Zhang, “Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems,” *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 450–455, Jun. 2014.

[19] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.

[20] Y. Wang and C. Dang, "An evolutionary algorithm for global optimization based on level-set evolution and latin squares," *IEEE Trans. Evol. Comput.*, vol. 11, no. 5, pp. 579–595, Oct. 2007.

[21] K. Bringmann, T. Friedrich, C. Igel, and T. Voí, "Speeding up many-objective optimization by Monte Carlo approximations," *Artif. Intell.*, vol. 204, no. 9, pp. 22–29, 2013.

[22] F. Gu and Y.-M. Cheung, "Self-organizing map-based weight design for decomposition-based many-objective evolutionary algorithm," *IEEE Trans. Evol. Comput.*, vol. 22, no. 2, pp. 211–225, Apr. 2018.

[23] Y.-M. Cheung, F. Gu, and H.-L. Liu, "Objective extraction for many-objective optimization problems: Algorithm and test problems," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 755–772, Oct. 2016.

[24] H.-L. Liu, F.-Q. Gu, and Y.-M. Cheung, "T-MOEA/D: MOEA/D with objective transform in multi-objective problems," in *Proc. Int. Conf. Inf. Sci. Manag. Eng. (ISME)*, 2010, pp. 282–285.

[25] Q. Yang *et al.*, "Adaptive multimodal continuous ant colony optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 2, pp. 191–205, Apr. 2017.

[26] M. Mavrovouniotis, F. M. Müller, and S. Yang, "Ant colony optimization with local search for dynamic traveling salesman problems," *IEEE Trans. Cybern.*, vol. 47, no. 7, pp. 1743–1756, Jul. 2017.

[27] F. Van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 225–239, Jun. 2004.

[28] L. Xu *et al.*, "Greedy criterion in orthogonal greedy learning," *IEEE Trans. Cybern.*, vol. 48, no. 3, pp. 955–966, Mar. 2018.

[29] V. A. Shim, K. C. Tan, and H. Tang, "Adaptive memetic computing for evolutionary multiobjective optimization," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 610–621, Apr. 2015.

[30] G. Zhang and Y. Li, "A memetic algorithm for global optimization of multimodal nonseparable problems," *IEEE Trans. Cybern.*, vol. 46, no. 6, pp. 1375–1387, Jun. 2016.

[31] J. Luo and F. Gu, "An adaptive niching-based evolutionary algorithm for optimizing multi-modal function," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 30, no. 3, 2016, Art. no. 1659007.

[32] X.-G. Zhou and G.-J. Zhang, "Abstract convex underestimation assisted multistage differential evolution," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2730–2741, Sep. 2017.

[33] Y. Mei, K. Tang, and X. Yao, "Decomposition-based memetic algorithm for multiobjective capacitated arc routing problem," *IEEE Trans. Evol. Comput.*, vol. 15, no. 2, pp. 151–165, Apr. 2011.

[34] K. Miettinen, *Nonlinear Multiobjective Optimization*. Norwell, MA, USA: Kluwer, 1999.

[35] Y.-M. Cheung and F. Gu, "On solving complex optimization problems with objective decomposition," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, 2013, pp. 2264–2269.

[36] R. Hooke and T. A. Jeeves, "'Direct search' solution of numerical and statistical problems," *J. ACM*, vol. 8, no. 2, pp. 212–229, 1961.

[37] M. J. D. Powell, "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *Comput. J.*, vol. 7, no. 2, pp. 155–162, 1964.

[38] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. New York, NY, USA: Wiley, 1999.

[39] Y.-M. Cheung and F. Gu, "A direct search algorithm based on kernel density estimator for nonlinear optimization," in *Proc. 10th Int. Conf. Nat. Comput.*, 2014, pp. 297–302.

[40] F. J. Solis and R. J.-B. Wets, "Minimization by random search techniques," *Math. Oper. Res.*, vol. 6, no. 1, pp. 19–30, 1981.

[41] Y. Yu and Z.-H. Zhou, "A new approach to estimating the expected first hitting time of evolutionary algorithms," *Artif. Intell.*, vol. 172, no. 15, pp. 1809–1832, 2008.

[42] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Stat. Soc. Series B (Methodol.)*, vol. 58, no. 1, pp. 267–288, 1996.

[43] X. Zong-Ben, G. Hai-Liang, W. Yao, and Z. Hai, "Representative of $l_{1/2}$ regularization among $l_q(0 \leq q \leq 1)$ regularizations: An experimental study based on phase diagram," *Acta Automatica Sinica*, vol. 38, no. 7, pp. 1225–1228, 2012.

[44] A. I. F. Vaz and L. N. Vicente, "A particle swarm pattern search method for bound constrained global optimization," *J. Glob. Optim.*, vol. 39, no. 2, pp. 197–219, 2007.

[45] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

Yiu-ming Cheung (SM'06–F'18) received the Ph.D. degree in machine learning from the Department of Computer Science and Engineering, the Chinese University of Hong Kong, Hong Kong.

He is currently a Full Professor with the Department of Computer Science, Hong Kong Baptist University, Hong Kong. His current research interests include machine learning, pattern recognition, visual computing, and optimization.

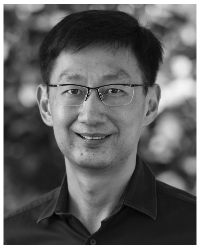
Dr. Cheung serves as an Associate Editor for several journals, including IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS and *Pattern Recognition*. He is also an IET Fellow, a BCS Fellow, an RSA Fellow, and an IETI Distinguished Fellow. More details can be found at: http://www.comp.hkbu.edu.hk/∽ymc.

Fangqing Gu received the B.S. degree from Changchun University, Changchun, China, in 2007, the M.S. degree from the Guangdong University of Technology, Guangzhou, China, in 2011, and the Ph.D. degree from the Department of Computer Science, Hong Kong Baptist University, Hong Kong, in 2016.

He joined the School of Applied Mathematics, Guangdong University of Technology as a Lecturer. His current research interests include data mining, machine learning, and evolutionary computation.

Hai-Lin Liu received the B.S. degree in mathematics from Henan Normal University, Xinxiang, China, the M.S. degree in applied mathematics from Xidian University, Xi'an, China, and the Ph.D. degree in control theory and engineering from the South China University of Technology, Guangzhou, China, and Postdoctoral degree from the Institute of Electronic and Information, South China University of Technology, Guangzhou.

He is currently a Professor with the School of Applied Mathematics, Guangdong University of Technology, Guangzhou, China. His research interests include evolutionary computation and optimization, and blind source separation.



Kay Chen Tan (SM'08–F'14) received the B.Eng. degree (First Class Hons.) in electronics and electrical engineering and the Ph.D. degree in evolutionary computation and control systems from the University of Glasgow, Glasgow, U.K., in 1994 and 1997, respectively.

He is a Full Professor with the Department of Computer Science, City University of Hong Kong, Hong Kong. He has published over 200 refereed articles and six books.

Dr. Tan is the Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, was the Editor-in-Chief of the *IEEE Computational Intelligence Magazine* from 2010 to 2013, and currently serves as the Editorial Board Member of over ten journals. He is an Elected Member of the IEEE CIS AdCom from 2017 to 2019.

Han Huang (M'15) received the B.Man. degree in information management and information systems from the School of Mathematics, South China University of Technology (SCUT), Guangzhou, China, in 2003 and the Ph.D. degree in computer science from SCUT in 2008.

He is currently a Professor with the School of Software Engineering, SCUT. His current research interests include evolutionary computation and swarm intelligence and their applications.

Dr. Huang is a Senior Member of CCF.