Water Flow Optimizer: A Nature-Inspired Evolutionary Algorithm for Global Optimization

Kaiping Luo

***Abstract*—Inspired by the shape of water flow in nature, a novel algorithm for global optimization, water flow optimizer (WFO), is proposed. The optimizer simulates the hydraulic phenomena of water particles flowing from highland to lowland through two operators: 1) laminar and 2) turbulent. The mathematical model of the proposed optimizer is first built, and then its implementation is described in detail. Its convergence is strictly proved based on the limit theory. The parametric effect is investigated. The performance of the proposed optimizer is compared with that of the related metaheuristics on an open test suite. The experimental results indicate that the proposed optimizer achieves competitive performance. The proposed optimizer was also successfully applied to solve the spacecraft trajectory optimization problem.**

***Index Terms*—Evolutionary computation, optimization, water flow optimizer (WFO).**

I. Introduction

NATURE-INSPIRED computational intelligence [1], [2] is becoming increasingly popular due to its intrinsic advantages: problem independency, derivative-free requirement, and simplicity. Various nature-inspired optimization algorithms have been developed since the genetic algorithm (GA) [3] was proposed. Nature-inspired metaheuristics can be classified into two categories. One is inspired by biological behaviors in nature, for example, particle swarm optimization (PSO) [4]–[7], ant colony optimization (ACO) [8]–[10], artificial bee colony (ABC) [11]–[13], firefly algorithm (FA) [14], [15], grey wolf optimizer (GSA) [16]–[18], and social spider algorithm (SSA) [19]–[21]. The other is completely based on some natural phenomena, such as biogeography-based optimization (BBO) [22], gravitational search algorithm (GSA) [23], intelligent water drops (IWDs) [24], and water wave optimization (WWO) [25]. However, there is no particular metaheuristic that can absolutely achieve the best solution for all optimization problems [26]. Consequently, it is still necessary to improve the existing algorithms and develop new metaheuristics.

Manuscript received 11 February 2020; revised 1 June 2020, 23 August 2020, and 25 November 2020; accepted 3 January 2021. Date of publication 10 February 2021; date of current version 19 July 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 71871004 and Grant 91646110. This article was recommended by Associate Editor K. Su.

The author is with the School of Economics and Management, Beihang University, Beijing 100191, China (e-mail: kaipingluo@buaa.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2021.3049607.

Digital Object Identifier 10.1109/TCYB.2021.3049607

In this article, a new algorithm for global optimization, water flow optimizer (WFO), is proposed. The WFO algorithm captures the shapes of water flow in nature: laminar and turbulent. In consequence, the WFO algorithm has two evolutionary operators of the same names: 1) laminar and 2) turbulent. By the two operators, the WFO algorithm mimics the hydraulic phenomena of water particles flowing from highland to lowland. Although algorithms IWD [24] and WWO [25] are also motivated by some phenomena of water flow in nature, they are different. The IWD algorithm uses some water drops to characterize the path selection of water flow in a natural river. In the IWD algorithm, a water drop is endued with an intelligence that enables it to select a path with less soil. The WWO algorithm simulates the phenomena of shallow water waves in the ocean, including propagation, refraction, and breaking. The WFO algorithm mimics the laminar and turbulent flow of water particles, including an interesting turbulent phenomenon, eddying, which is often observed in rivers and even near the bottom orifice of a bathtub.

The theoretical contributions of this study to the design of metaheuristics, including swarm intelligence and evolutionary computation, are mainly reflected in the following three aspects.

First, a new integrated search framework is proposed. To the best of the author's knowledge, the vectorial search pattern is employed by almost all population-based metaheuristics except for the ABC algorithm. The vector-based multidimensional search pattern has a high efficiency in solving separable functions due to its simultaneous search property in all dimensions. However, its efficiency is often lower than that of the unidimensional search pattern in solving nonseparable functions. Compared with the vectorial search pattern, the unidimensional search pattern easily leads to a slow convergence rate, but can well maintain the structure hidden in better solutions. The WFO algorithm takes full their advantages by integrating. More specifically, in the WFO algorithm, the laminar operator adopts a vectorial search pattern while the turbulent operator uses a unidimensional search pattern.

Second, a novel approach to avoiding local optima is proposed. To the best of the author's knowledge, almost all existing metaheuristics adopt the nonparallel multidirectional search pattern so that they have to resort to more sophisticated control parameters or a special operation for exploration, for example, mutation in GA and differential evolution (DE) and reinitialization in artificial colony bee and harmony search [27], [28]. In the WFO algorithm, a distinctive approach is proposed to avoid local optima. More specifically, the laminar operator in the WFO algorithm compels each agent

to search the solution space in parallel by a common direction during each iteration. By the parallel uni directional search approach, many agents are directly moving toward a better guiding solution while other agents are practically searching backwards in the same iteration. As a result, each search agent has an equal chance of jumping over local optima.

Third, an effective methodology for proving algorithmic convergence is proposed based on the limit theory. The theoretical convergence of a metaheuristic is still an open intractable issue, even though a few studies [29]–[33] have been devoted to this aspect. However, most address the discrete solution space using the Markov chains and analyze the convergence under some particular cases. In this article, the convergence of an algorithm is equivalently transformed into the limit of the solution sequence yielded by the algorithm in each dimension. Such a methodology effectively proves the convergence of the WFO algorithm and the global convergence in solving the unimodal functions.

The remainder of this article is organized as follows. The next section presents the phenomena of water flow in hydraulics, and then the simulated model is built from an optimization viewpoint. Section III states the details of the algorithm's implementation. In Section IV, the convergence of the WFO algorithm is strictly proved by the limit theory. Section V reports the numerical experimental results. Real application in engineering is presented in Section VI. Section VII highlights conclusions and provides several suggestions for future research.

II. Water Flow Optimizer

A. Water Flow in Hydraulics

There are two distinctly different types of water flow in nature: 1) laminar and 2) turbulent. In 1883, Osborne Reynolds demonstrated the two types of water flow through a special experiment [34]. In the experiment, a thread-like stream of colored liquid with the same density as water was injected into a transparent glass tube with a valve at the end. At the same time, water continually flowed through the glass tube to a tank. When the valve was slowly opened, the velocity of water flow in the tube was small and the colored liquid flowed as a straight line throughout the length of the tube. This regular phenomenon is defined as *laminar flow* in hydraulics. As the valve was further opened, the velocity of water in the tube gradually increased. Up to a certain velocity, the colored streamline first became wavy and then uniformly diffused such that numerous vortices emerged and no streamline could be distinguished. This irregular phenomenon is defined as *turbulent flow* in hydraulics. If the valve was slowly closed, the colored streamline would reappear as the water flow slowed down.

In short, when the velocity of water flow is small, water particles distributed on different layers regularly move in parallel straight lines, as shown in Fig. 1(a). Such a regular flow is called laminar flow. When the velocity of water flow is large, water particles jostle each other in a stochastic manner and move over layers, as illustrated in Fig. 1(b). Such an irregular flow is known as turbulent flow. In the real world, we may

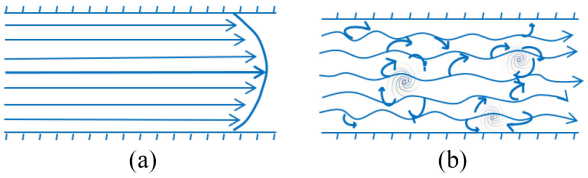

Fig. 1. Shapes of water flow. (a) Laminar flow. (b) Turbulent flow.

encounter laminar flow in open channels or troughs. More commonly, laminar flow occurs as sheet flow, a thin sheet of flowing liquid, such as that in drainage from sidewalks, streets, and airport runways. In rivers, water flow is mostly turbulent.

In hydraulics, water flow can be identified by the dimensionless Reynolds number, which is defined as the ratio of inertial forces to viscous forces. In general, water flow is laminar if its Reynolds number is less than a threshold and is turbulent otherwise [34]. The Reynolds number indicates that velocity is not the single factor that determines whether the flow is laminar or turbulent. The irregularity of turbulence is essentially derived from the effective mixing action of water particles between adjacent layers. A rapidly moving molecule shifting into a slower moving layer tends to speed up the latter. Likewise, a slowly moving molecule entering a faster moving layer tends to slow down the latter. This molecular interchange sets up a shear, or produces a friction force between adjacent layers. Resistance to shear is known as viscosity in fluid mechanics [35]. On boundaries, water particles adhere to the wall and the inner friction force almost disappears. As a result, the velocity of water particles at boundaries is equal to 0 relative to the wall. The velocity of water particles away from boundaries becomes gradually large as the shearing stress increases. Thus, the velocity profile is a parabolic curve in the sectional view. As shown in Fig. 1(a), the velocities of water particles are maximum at the center line of the glass tube. When the rapid water flow is interfered with by obstacles, local oscillation will inevitably occur. Both velocity and pressure are also changed. In the vicinity of a wave peak, the velocity of water particles above the peak increases and the pressure reduces because the cross section decreases. On the contrary, the velocity of water particles below the peak reduces and the pressure increases due to the enlargement of the cross section. Likewise, the velocity of water particles adjacent to the valley and the pressure are changed. But, the fluctuant tendency is just inverse. In fluctuation, the shearing force tends to swirl water particles as a torque. Consequently, an upward force acts in a peak while a downward force acts in a valley. The shape of a peak becomes more convex, and the shape of a valley becomes more concave. The fluctuation becomes consequently clearer. Once the peak and the valley overlap, a resultant eddy appears.

B. Mathematical Model of Water Flow Optimizer

Laminar and turbulent flow in nature give us good inspiration for designing regular and irregular stochastic search operators in optimization. Therefore, the two types of water flow can be modeled from an optimization viewpoint. Without loss of generality, suppose that the problem to be solved has

an objective function to be minimized. The optimal solution can be found by a sufficiently and effectively iterative process from solutions with a big value of objective function to solutions with a small value of objective function. Since water is always flowing from a source at high altitude to a sink at low altitude in nature, there exists a similarity between water flow and solution search. More specifically, if a water particle is viewed as a solution, the particle's position is analogous to the solution's value, and the potential energy of the water particle is similar to the objective or fitness. Consequently, the objective function value for the problem to be minimized or the inverse for the problem to be maximized is defined as the potential energy of a water particle. The objective function is directly used as the evaluation function of the potential energy of a water particle in the WFO algorithm.

1) Laminar Operator: In laminar flow, all particles move in parallel straight lines. Due to the viscosity of water, the velocities of particles may be different, for example, the particles in layers away from walls or obstacles are faster than those in layers close to walls or obstacles. This regular motion can be modeled by the following equation:

$$\boldsymbol{y}_i(t) = \boldsymbol{x}_i(t) + s * \vec{d} \quad \forall i \in \{1, 2, \ldots, m\} \tag{1}$$

where t is the iteration number, i and m are, respectively, the index and number of employed water particles, $\boldsymbol{x}_i(t)$ is the position of the ith particle at the t-th iteration, $\boldsymbol{y}_i(t)$ denotes its possible moving position, random number $s \in U([0, 1])$ represents the shifting coefficient of a water particle, and the $\vec{d}$ vector indicates a common motional direction for all particles. The common direction is determined by two distinct particles randomly selected as follows:

$$\vec{d} = \boldsymbol{x}_h(t) - \boldsymbol{x}_k(t), \ \ (h \neq k, f(\boldsymbol{x}_h(t)) \leq f(\boldsymbol{x}_k(t))) \tag{2}$$

where $f(\boldsymbol{x}_h(t)) \leq f(\boldsymbol{x}_k(t))$ denotes that the potential energy of the jth particle, $f(\boldsymbol{x}_h(t))$, is lower than or equal to that of the kth particle, $f(\boldsymbol{x}_k(t))$. In particular, if particle h is fixed at the best solution found so far, any other particle can be employed as particle k. The best solution found represents the position currently having the lowest potential energy. Thus, the best solution found so far is selected as the guiding particle h in implementing the laminar operator.

Equations (1) and (2) are similar in form to the simple mutation operator in DE [36]. It should be noted that the value of s in (1) varies from water particle to water particle, but the value of $\vec{d}$ is unchanged for each water particle during the same iteration. As a result, there exists an essential difference in terms of search pattern. The laminar operator in the WFO algorithm adopts a regular parallel unidirectional search pattern while the mutation operator in the DE algorithm uses a chaotic multidirectional search pattern. In (1), the randomness of s ensures that the different water particles have unequal shifts and the commonness of $\vec{d}$ guarantees a parallel unidirectional search.

2) Turbulent Operator: In the turbulent flow, water particles jostle each other in a stochastic manner and move over adjacent layers. The streamline is broken due to the irregular motion of water particles. When the rapid water flow is interfered with by obstacles, local oscillation will occur. Once the amplitude of oscillation is sufficiently large, the shearing force, as a torque, impels water particles to swirl. As a result, an eddy appears. If a dimension of the problem to be solved is regarded as a layer of water flow, a mathematical transformation between dimensions can simulate the irregular motion of water particles in turbulent flow. Naturally, the possible moving position of a water particle in the turbulent operator, $\boldsymbol{y}_i$, is generated by oscillating in a randomly selected dimension, that is

$$\boldsymbol{y}_i(t) = \left(\ldots, x_i^{j1-1}(t), v, x_i^{j1+1}(t), \ldots\right) \tag{3}$$

where $j1 \in \{1, 2, \ldots, n\}$ is a component randomly selected from n dimensions, and the mutation value v is determined by the following equation:

$$v = \begin{cases} \psi\left(x_i^{j1}(t), x_k^{j1}(t)\right), & \text{if } r < p_e \\ \varphi\left(x_i^{j1}(t), x_k^{j2}(t)\right), & \text{otherwise} \end{cases} \tag{4}$$

where $k \in \{1, 2, \ldots, m\}$ is the index of the randomly selected distinct particle, $k \neq i$; $j2$ is a randomly selected distinct dimension, $j2 \neq j1$, r is a random number in the range [0, 1] and $p_e \in (0, 1)$ is a control parameter called the eddying probability. Namely, the turbulent flow operator in the WFO algorithm implements the interesting eddying transformation ψ with the probability of p_e and the general over-layer moving transformation φ with the probability of $1 - p_e$.

Because the shape of an eddy resembles a special Archimedean spiral, the eddying function in the turbulent operator is constructed by a similar equation

$$\psi\left(x_i^{j1}(t), x_k^{j1}(t)\right) = x_i^{j1}(t) + \rho * \theta * \cos(\theta) \tag{5}$$

where θ is a random number within the range $[-\pi, \pi]$, and ρ is dynamically determined by a self-adaptive rule

$$\rho = \left|x_i^{j1}(t) - x_k^{j1}(t)\right|. \tag{6}$$

ρ can be viewed as the shearing force of the kth particle to the ith particle. Surely, it is also reasonable to use the basic sine function in (5) due to $\sin(\theta) = \cos(\theta - [\pi/2])$.

To emulate the general behavior of water particles over layers, the following transformation function is employed:

$$\varphi\left(x_i^{j1}(t), x_k^{j2}(t)\right) = \left(ub^{j1} - lb^{j1}\right) * \frac{x_k^{j2}(t) - lb^{j2}}{ub^{j2} - lb^{j2}} + lb^{j1} \tag{7}$$

where lb and ub, respectively, stand for the lower and upper bounds. The linear transformation can inherit the scale of a randomly selected particle to the bounds in a selected dimension.

3) Evolutionary Rule: In hydraulics, water flow is laminar if the Reynolds number is less than a threshold and turbulent otherwise. The two types of water flow are stochastically simulated in the WFO algorithm. More specifically, the laminar flow is mimicked with the probability of p_l and the turbulent flow is emulated with the probability of $1 - p_l$. The laminar probability $p_l \in (0, 1)$, as a key control parameter, is similar to the threshold of the Reynolds number because the value of

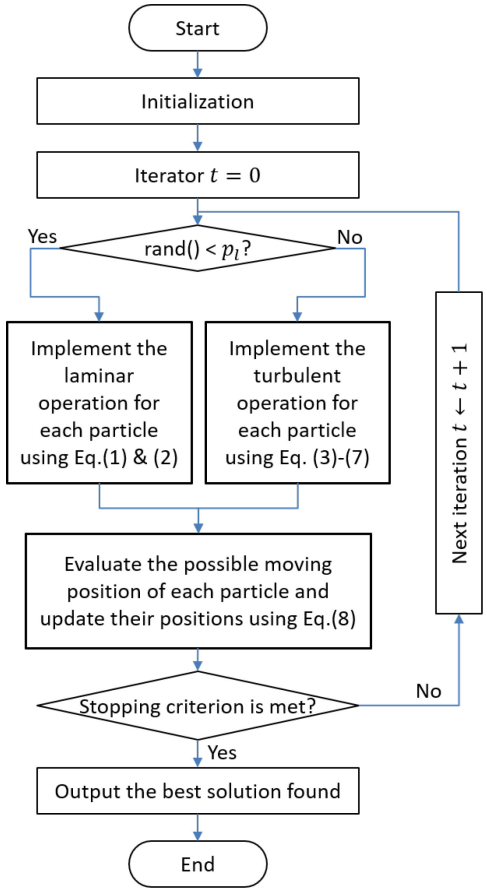

Fig. 2. Flowchart of the WFO.

p_l also determines which operation, laminar or turbulent, is carried out.

The objective is continually optimized over the course of iteration. Hence, the trend of water flow from highland to lowland is characterized in the WFO algorithm by a common strategy in evolutionary computation. More specifically, if the possible moving position has a lower potential energy, it will be selected as the new position; otherwise, the water particle remains at the current position. Namely

$$\boldsymbol{x}_i(t+1) = \begin{cases} \boldsymbol{y}_i(t), & \text{if } f\big(\boldsymbol{y}_i(t)\big) < f(\boldsymbol{x}_i(t)) \\ \boldsymbol{x}_i(t), & \text{otherwise.} \end{cases} \tag{8}$$

III. Algorithmic Implementation

It is pretty easy to implement the proposed optimizer according to the pseudo code shown in Algorithm 1. It starts from a random population of water particles and continually updates the position of each water particle using the laminar or turbulent operator until a given stopping criterion is satisfied, as shown in Fig. 2. The MATLAB toolbox of WFO is also available from the author's homepage (http://shi.buaa.edu.cn/luokaiping).

A. Initialization

In the initialization phase, it is necessary to assign values to the algorithmic parameters, including the number of employed water particles m, the laminar probability p_l, and the eddying probability p_e. Besides parameter settings, both the position and potential energy of each water particle are initialized in the first phase. The initial position of water particles is randomly generated within the search space. The potential energy of each particle is directly computed using the objective function.

B. Evolutionary Operations

The WFO algorithm generates a trial solution by the laminar or turbulent operator. It is easy to implement the two operators according to (1)–(8). However, the following points should be noted in implementing the laminar operator.

1) The value of s is distinct for different particles but identical on each component of position for the same water particle.
2) The value of $\vec{d}$ may be distinct at different iterations but identical for each water particle during the same iteration; to speed up the convergent rate, the best solution found so far is the optimal candidate of the guiding particle h.

C. Constraint Handling

The value of the trial solution produced by the laminar or turbulent operator may exceed the search bounds. There are few ways to resolve the issue of boundary constraint violation [37], such as random, absorbing and reflecting. In this study, the retaining scheme is used to handle the boundary constraint violation. More specifically, $y_i^j(t) = x_i^j(t)$, if $y_i^j(t) > ub^j$ or $y_i^j(t) < lb^j$. It should be noted that the value of the trial solution yielded by (7) does not exceed any bound. To reduce the unnecessary computing consumption; therefore, an operation for checking boundary constraints is implemented after generating a trial solution by (1) and (5).

There are also some approaches to handle the constraints of real-world engineering problems [18], [38], for example, the penalty function, decoder, special repair operator, and separation of objective function and constraints. If a repair operator is employed in solving the constrained problem, a special operation based on the knowledge of the problem should be embedded before the updating phase to boost the efficiency of the WFO algorithm.

IV. Algorithmic Convergence

Definition 1: If there exists a vector $\boldsymbol{x}^* \in \mathbb{R}^n$ such that $\lim_{t\to\infty} \|\boldsymbol{x}(t) - \boldsymbol{x}^*\| = 0$, we say that the vectorial sequence $\{\boldsymbol{x}(t)\}$ converges to $\boldsymbol{x}^*$ or tends to $\boldsymbol{x}^*$ by a given norm $\|\cdot\|$ and $\boldsymbol{x}^*$ is called the limit of the sequence.

Definition 2: A sequence $\{\boldsymbol{x}(t)\}$ is called a Cauchy sequence if $\lim_{t1,t2\to\infty} \|\boldsymbol{x}(t1) - \boldsymbol{x}(t2)\| = 0$.

Lemma 1 (Cauchy's Convergence Criterion [39]): $\lim_{t\to\infty} \|\boldsymbol{x}(t) - \boldsymbol{x}^*\| = 0$ with $\boldsymbol{x}^* \in \mathbb{R}^n$, iff $\lim_{t1,t2\to\infty} \|\boldsymbol{x}(t1) - \boldsymbol{x}(t2)\| = 0$.

Lemma 1 indicates that a sequence is convergent if and only if it is a Cauchy sequence. Let $t1 = t + 1$ and $t2 = t$. It is natural to deduce the following corollary by this lemma.

Corollary 1: If $\lim_{t\to\infty} \|\boldsymbol{x}(t+1) - \boldsymbol{x}(t)\| = 0$, the sequence $\{\boldsymbol{x}(t)\}$ is convergent.

Algorithm 1: WFO

foreach *water particle* $i \in \{1, 2, \ldots, m\}$ **do** /* Initialization */
  foreach *dimension* $j \in \{1, 2, \ldots, n\}$ **do**
    Randomly generate a position, $x_i^j = lb^j + rand() * (ub^j - lb^j)$.
  end
  Calculate the potential energy, $f_i = f(\boldsymbol{x}_i)$.
end
while *A given stopping criterion is Not satisfied.* **do**
  if $rand() < p_l$ **then** /* Laminar flow */
    Determine a laminar direction, $\vec{d} = \boldsymbol{x}_h - \boldsymbol{x}_k$.
    foreach *water particle* $i \in \{1, 2, \ldots, m\}$ **do**
      $s = rand()$ and $\mathbf{y}_i = \boldsymbol{x}_i + s * \vec{d}$.
    end
  else /* Turbulent flow */
    Initialize a trial position, $\mathbf{y}_i \leftarrow \boldsymbol{x}_i$.
    Randomly select a jostling particle, $k \in \{1, 2, \ldots, m\}$ and $k \neq i$.
    Randomly select a dimension, $j1 \in \{1, 2, \ldots, n\}$.
    if $rand() < p_e$, **then** /* Eddying */
      $\rho = |x_i^{j1} - x_k^{j1}|$, $\theta = 2 * \pi * rand() - \pi$ and $y_i^{j1} = x_i^{j1} + \rho * \theta * \cos(\theta)$.
    else /* Over-layer moving */
      Randomly select another dimension, $j2 \in \{1, 2, \ldots, n\}$ and $j2 \neq j1$. $y_i^{j1} = \left(ub^{j1} - lb^{j1}\right) * \frac{x_k^{j2} - lb^{j2}}{ub^{j2} - lb^{j2}} + lb^{j1}$.
    end
  end
  foreach *water particle* $i \in \{1, 2, \ldots, m\}$ **do** /* Evolving and updating */
    if $f(\mathbf{y}_i) < f_i$ **then**
      $f_i \leftarrow f(\mathbf{y}_i)$ and $\boldsymbol{x}_i \leftarrow \mathbf{y}_i$.
      if $f_i < f_b$ **then**
        $f_b \leftarrow f_i$, $\boldsymbol{x}_b \leftarrow \boldsymbol{x}_i$ and $b \leftarrow i$.
      end
    end
  end
end

Lemma 2: $\lim_{t1,t2\to\infty} \|\boldsymbol{x}(t1) - \boldsymbol{x}(t2)\| = 0$ iff $\forall j \in \{1, 2, \ldots, n\}$, $\lim_{t1,t2\to\infty} |x^j(t1) - x^j(t2)| = 0$.

Lemma 2 indicates that a vectorial sequence converges if only if it is convergent in each dimension. It is easy to prove Lemma 2 by the definition of limit and norm.

Lemma 3 (Two-Sided Clamping Criterion [39]): If there are three sequences $\{x(t)\}$, $\{y(t)\}$, and $\{z(t)\}$ such that $\lim_{t\to\infty} y(t) = \lim_{t\to\infty} z(t) = l$ and $\forall t \in \{1, 2, \ldots\}$, $y(t) \leq x(t) \leq z(t)$, then $\lim_{t\to\infty} x(t) = l$.

It is also easy to deduce the following corollary by Lemma 3.

Corollary 2: If $\forall t \in \{1, 2, \ldots\}$, $0 \leq x(t) \leq z(t)$, and $\lim_{t\to\infty} z(t) = 0$, then $\lim_{t\to\infty} x(t) = 0$.

Definition 3: If the best solution sequence found by an algorithm is convergent, we say that the algorithm converges.

Theorem 1: Let $\boldsymbol{x}_i(t)$ be the position of water particle i during the t-th iteration. Then, $\lim_{t\to\infty} \|\boldsymbol{x}_i(t+1) - \boldsymbol{x}_i(t)\| = 0$, $i \in \{1, 2, \ldots, m\}$.

Proof: The vectorial sequence $\{\boldsymbol{x}_i(t)\}$ converges as long as the scalar sequence $\{x_i^j(t)\}$ is convergent in each dimension $j \in \{1, 2, \ldots, n\}$ by Corollary 1 and Lemma 2. By Corollary 2, we do only prove $\lim_{t\to\infty} \max_i\{|x_i^j(t+1) - x_i^j(t)|\} = 0$ due to $|x_i^j(t+1) - x_i^j(t)| \leq \max_i\{|x_i^j(t+1) - x_i^j(t)|\}$.

By (1) and (2), $y_i^j(t) \in o(x_i^j(t), \delta_l(t))$, where $\delta_l(t) = s * |x_h^j(t) - x_k^j(t)|$. Evidently, $\delta_l(t) \leq \max_j\{\max_{h,k}\{|x_h^j(t) - x_k^j(t)|\}\}$.

By (5) and (6), $y_i^{j1}(t) \in o(x_i^{j1}(t), \delta_e(t))$, where $\delta_e(t) = \pi * |x_i^{j1}(t) - x_k^{j1}(t)|$. Obviously, $\delta_e(t) \leq \pi * \max_j\{\max_{i,k}\{|x_i^j(t) - x_k^j(t)|\}\}$.

By (7), $|y_i^{j1}(t) - x_i^{j1}(t)| = [1/(ub^{j2} - lb^{j2})]|(ub^{j1} - lb^{j1}) * x_k^{j2}(t) - (ub^{j2} - lb^{j2}) * x_i^{j1}(t) + lb^{j1} * ub^{j2} - lb^{j2} * ub^{j1}|$. In particular, $|y_i^{j1}(t) - x_i^{j1}(t)| = |x_k^{j2}(t) - x_i^{j1}(t)|$, if $\forall j1, j2 \in \{1, 2, \ldots, n\}$, $ub^{j1} = ub^{j2}$ and $lb^{j1} = lb^{j2}$. It should be noted that the expression of (7) is deterministic because there is not a random number in the equation. The determinateness implies that the number of trial solution is equal to the dimensional number of the problem to be solved and the over-layer moving operation randomly examines one from the set containing the limited trial solutions once the jostling particles i and k are determined. In other words, an algorithm will soon converge if it only utilizes the same transformation as (7) because it only examines the limited solutions scattered in the entire search space. In WFO, the over-layer moving operation presented by (7) is carried out with probability $1 - p_e$. The stochastic simulation guarantees that the over-layer moving operation can transform more distinct solutions because of the change of $\boldsymbol{x}_i(t)$. In essence, the over-layer moving operation speeds up rather than hinders the convergence rate of the WFO. Therefore, it is reasonable to omit the over-layer moving operation hereafter because the theorem focuses on convergence rather than the convergent rate.

By (8) $\forall i \in \{1, 2, \ldots, m\}, j \in \{1, 2, \ldots, n\}$, $x_i^j(t+1) = y_i^j(t)$, if $f(\mathbf{y}_i(t)) < f(\boldsymbol{x}_i(t))$; otherwise, $x_i^j(t+1) = x_i^j(t)$.

For conciseness

$$\alpha_j(t) \triangleq \max_i \left\{ \left| x_i^j(t+1) - x_i^j(t) \right| \right\} \tag{9}$$

and

$$\beta(t) \triangleq \max_j \left\{ \max_{h,k} \left\{ \left| x_h^j(t) - x_k^j(t) \right| \right\} \right\}. \tag{10}$$

Notably, $\max_j\{\max_{h,k}\{|x_h^j(t) - x_k^j(t)|\} = \max_j\{\max_{i,k}\{|x_i^j(t) - x_k^j(t)|\}$ because of $h \in \{1, 2, \ldots, m\}$ and $i \in \{1, 2, \ldots, m\}$. Thus $\forall j \in \{1, 2, \ldots, n\}$, $\alpha_j(t) \leq \beta(t)$ in the laminar operation and $\alpha_j(t) \leq \pi * \beta(t)$ in the eddying operation.

Next, we discuss the limit of sequence $\{\alpha_j(t)\}$ under three cases regarding the convergence of sequence $\{\beta(t)\}$.

1) If $\lim_{t\to\infty}\beta(t) = 0$, then $\lim_{t\to\infty}\alpha_j(t) = 0$ by Corollary 2.
2) If $\lim_{t\to\infty}\beta(t) = l$ but $l \neq 0$, then there exists $T > 0$ such that $\alpha_j(t) \equiv 0$ if $t > T$, and the problem to be solved has at least two optimal solutions. Suppose $h = h0$ and $k = k0$ such that $|x^j_{h0}(t) - x^j_{k0}(t)| = \max_j\{\max_{h,k}\{|x^j_h(t) - x^j_k(t)|\}\}$. In other words, the distance between particle $h0$ and particle $k0$ is the farthest during the t-th iteration. $\lim_{t\to\infty}\beta(t) \neq 0$ indicates that there exists $T > 0$ such that the value of $\beta(t)$ is constant if $t > T$. Otherwise, there certainly exists a better solution in a neighborhood of particle $h0$ or $k0$ by the evolutionary rule of the WFO. The value of $\beta(t)$ will not be changed until particles $h0$ and $k0$ are severally located at two distinct optimal solutions, namely $\forall \boldsymbol{y} \in o(\boldsymbol{x}_{h0}(t), \beta(t))$, $\boldsymbol{y} \neq \boldsymbol{x}_{h0}(t)$, or $\forall \boldsymbol{y} \in o(\boldsymbol{x}_{k0}(t), \beta(t))$, $\boldsymbol{y} \neq \boldsymbol{x}_{k0}(t)$, and then $f(\boldsymbol{y}) > f(\boldsymbol{x}_{h0}(t)) = f(\boldsymbol{x}_{k0}(t))$. In other words, $\forall i \in \{1, 2, \ldots, m\}$, $\boldsymbol{x}_i(t) = \boldsymbol{x}_{h0}(t)$ or $\boldsymbol{x}_i(t) = \boldsymbol{x}_{k0}(t)$, where $t > T$.
3) If the sequence $\{\beta(t)\}$ diverges, a contradiction will occur. If we assume that the sequence $\beta(t)$ diverges, then there exists a sufficiently large number $T > 0$ such that the value of $\beta(t)$ is still oscillatory when $t > T$. The oscillation implies that particle $h0$ or $k0$ is endlessly replaced by a better neighbor solution that is sometimes close to and sometimes far away from the replacement solution. This is contradictory with the rigorous evolutionary rule of the WFO. ■

Theorem 1 indicates that the solution sequence yielded by each search agent in the WFO is convergent. It should be noted that these solution sequences may converge to different optimal solutions for the multimodal problem according to the above proof. Because the recorded best solution is the best one among each generation population, it is natural to obtain the following corollary by Theorem 1.

Corollary 3: Let $\{\boldsymbol{x}_b(t)\}$ be the best solution sequence found by the WFO. $\lim_{t\to\infty}\|\boldsymbol{x}_b(t+1) - \boldsymbol{x}_b(t)\| = 0$.

Corollary 3 indicates that the WFO is convergent.

Definition 4: $\boldsymbol{x}^* \in \mathbb{R}^n$ is a global optimal solution of the evaluation function $f(\cdot)$ to be minimized if $\forall\delta > 0$ $\forall \boldsymbol{x} \in o(\boldsymbol{x}^*, \delta), \boldsymbol{x} \neq \boldsymbol{x}^*, f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*)$. $\boldsymbol{x}^* \in \mathbb{R}^n$ is a local optimal solution if $\exists\delta > 0$ $\forall \boldsymbol{x} \in o(\boldsymbol{x}^*, \delta)$, $\boldsymbol{x} \neq \boldsymbol{x}^*, f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*)$.

Definition 5: A stochastic search algorithm is said to be globally convergent if the best solution sequence found by the stochastic search algorithm tends to a global optimal solution. A stochastic search algorithm is said to be locally convergent if the best solution sequence found by the stochastic search algorithm tends to a local optimal solution.

Evidently, a globally convergent algorithm is locally convergent, but the contrary is not true.

Theorem 2: Let $\{\boldsymbol{x}_b(t)\}$ be the best solution sequence found by the WFO and $\boldsymbol{x}^*_g \in \mathbb{R}^n$ be a global optimal solution of the evaluation function $f(\cdot)$. If $\exists T > 0$ $\forall t > T$, $\exists i \in \{1, 2, \ldots, m\}$, $\boldsymbol{x}_i(t) \in o(\boldsymbol{x}^*_g, \max_j\{\max_{h,k}\{|x^j_h(t) - x^j_k(t)|\})$, then $\lim_{t\to\infty} f(\boldsymbol{x}_b(t)) = f(\boldsymbol{x}^*_g)$.

Proof: For conciseness, we continue to use the symbol $\beta(t)$ instead of $\max_j\{\max_{h,k}\{|x^j_h(t) - x^j_k(t)|\}$.

If $\exists T > 0$ $\forall t > T$, $\exists i \in \{1, 2, \ldots, m\}$, $\boldsymbol{x}_i(t) \in o(\boldsymbol{x}^*_g, \beta(t))$, then we can select the vector $\boldsymbol{x}_i(t)$ after the T-th iteration to artificially assemble a new sequence $\{\boldsymbol{z}(t)\}$. Naturally, $\boldsymbol{z}(t)$ inherits the properties of such a vector $\boldsymbol{x}_i(t)$, as follows:

$$f\left(\boldsymbol{x}^*_g\right) \leq f(\boldsymbol{x}_b(t)) \leq f(\boldsymbol{z}(t))$$

and

$$\left\|\boldsymbol{z}(t) - \boldsymbol{x}^*_g\right\| \leq \beta(t).$$

1) If $\lim_{t\to\infty}\beta(t) = 0$, then $\lim_{t\to\infty}\|\boldsymbol{z}(t) - \boldsymbol{x}^*_g\| = 0$ by Corollary 2. $\lim_{t\to\infty} f(\boldsymbol{x}_b(t)) = f(\boldsymbol{x}^*_g)$ by Lemma 3.
2) If $\lim_{t\to\infty}\beta(t) > 0$, there exists $T' > 0$ such that the value of $\beta(t)$ is constant if $t > T'$. At the same time, there exists other distinct global optimal solution $\boldsymbol{x}^\#_g \in o(\boldsymbol{x}^*_g, \beta(t))$ such that $\|\boldsymbol{x}^\#_g - \boldsymbol{x}^*_g\| = \lim_{t\to\infty}\beta(t)$ and $f(\boldsymbol{x}^\#_g) = f(\boldsymbol{x}^*_g)$. As a result $\forall i \in \{1, 2, \ldots, m\}, \boldsymbol{x}_i(t) = \boldsymbol{x}^\#_g$ or $\boldsymbol{x}_i(t) = \boldsymbol{x}^*_g$, where $t > T'$. Namely $\forall i \in \{1, 2, \ldots, m\}$, $\lim_{t\to\infty} f(\boldsymbol{x}_i(t)) = f(\boldsymbol{x}^\#_g) = f(\boldsymbol{x}^*_g)$. Because $f(\boldsymbol{x}_b(t)) = \min_i f(\boldsymbol{x}_i(t))$, we have $\lim_{t\to\infty} f(\boldsymbol{x}_b(t)) = f(\boldsymbol{x}^*_g)$.
3) If the sequence $\{\beta(t)\}$ diverges, the same contradiction as mentioned in the proof of Theorem 1 will occur. ■

Theorem 2 indicates that the WFO globally converges if there exists a solution yielded by it in each iteration such that the distance from the solution to a global optimal solution is always less than $\beta(t)$ after a sufficiently large number of iteration. For the unimodal function, $\lim_{t\to\infty}\beta(t) = 0$, because the condition in Theorem 2 is satisfied for the unimodal function in light of the evolutionary scheme of the WFO. Therefore, we can deduce a stricter corollary.

Corollary 4: Let $\{\boldsymbol{x}_b(t)\}$ be the best solution sequence found by the WFO and $\boldsymbol{x}^*_g \in \mathbb{R}^n$ be the global optimal solution of the unimodal function. $\lim_{t\to\infty}\boldsymbol{x}_b(t) = \boldsymbol{x}^*_g$.

Corollary 4 indicates that the WFO certainly can steadily converge to the single global optimal solution of the unimodal function.

For the multimodal function, the condition in Theorem 2 may be not satisfied. However, we can raise the globally convergent probability of the WFO

$$p = \prod_{t>T} p^t\left\{\boldsymbol{x}^*_g \in \bigcup_i o(x_i(t), \delta_i(t))\right\} \tag{11}$$

where $\delta_i(t)$ is the flowing radius of the ith water particle in the t-th iteration and $\delta_i(t) \leq \pi * \beta(t)$. In the early stage of the iteration, the value of $\delta_i(t)$ is relatively large so that the global optimal solution $\boldsymbol{x}^*_g$ is located in the attainable search range $\bigcup_i o(x_i(t), \delta_i(t))$. Due to the self-adaptiveness of $\delta_i(t)$, its value continually changes and ultimately reduces to a constant, which may be equal to 0. In the later stage of the iteration, the global optimal solution $\boldsymbol{x}^*_g$ may be out the attainable search range for the multimodal function if the flowing radius of each water particle is too small. Hence, it is crucial for the global optimal solution to be continuously located in an attainable neighborhood after a sufficiently large number of iteration $T > 0$, rather than over the entire course of iteration. If each

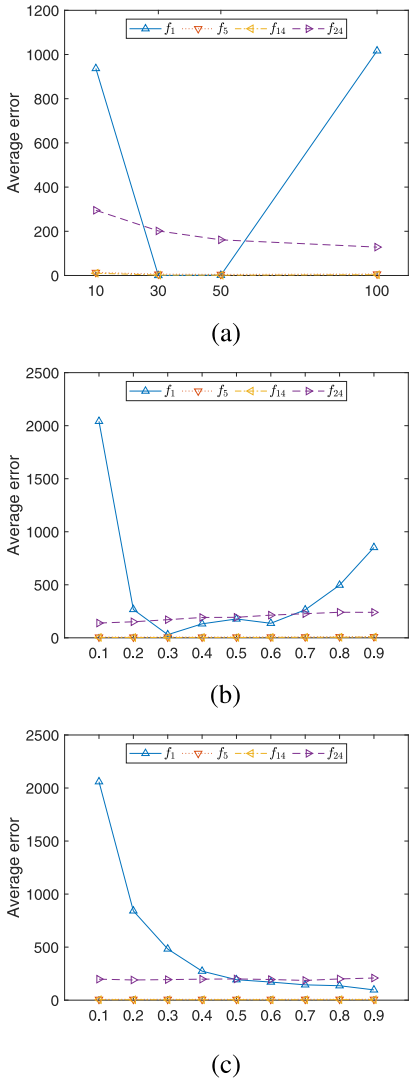

Fig. 3. Average error under the different parameter settings. (a) Population size, m. (b) Laminar probability, p_l. (c) Eddying probability, p_e.

water particle maintains a large flowing radius, global convergence is guaranteed in theory, but the convergent rate will become extremely slow, even stagnant because a large number of inferior solutions will be examined and even repeatedly searched during the limited number of iterations.

V. Algorithmic Performance

A. Benchmark Problems and Experimental Environment

In this study, the open CEC2017 test suite [40] was employed. The latest test suite was generated by shifting, rotating, and hybridizing well-known benchmark problems. In consequence, the open test suite is composed of three unimodal functions, seven simple multimodal functions, ten hybrid functions, and ten composition functions.

In the experiment, all algorithms were coded in MATLAB and executed on a computer with eight 3.0-GHz CPUs and 64-GB of RAM.

B. Effect of Parameters

The WFO algorithm has two key control parameters: 1) laminar probability p_l and 2) eddying probability p_e. To investigate the effect of the two parameters as well as the population size m on the performance of the WFO algorithm, an orthogonal experiment was performed. In the experiment, p_l, p_e, and m are viewed as three factors with 9, 9, and 4 levels, respectively. Under these different parameter settings, the WFO algorithm repeatedly solved four 10-D representatives in the CEC2017 test: 1) unimodal function $f1$; 2) simple multimodal function $f5$; 3) hybrid function $f14$; and 4) composition function $f24$.

Fig. 3 shows the average error of four representative functions over 30 independent runs. It is seen that $m \in [20, 50]$, $p_l \in [0.2, 0.5]$, and $p_e \in [0.5, 0.9]$ can achieve a relatively smaller average error for these test functions.

Table I summarizes the P-values of the multiway analysis of variance for testing the effect of parameters on the performance of the WFO algorithm in solving the above representative functions. As shown in the table, for function $f1$, those P-values are more than 0.01, except for that of the interaction between m and p_l. This just verifies that the fluctuation of the average error of function $f1$ is the most obvious in Fig. 3. Compared with the other three functions, the performance of the unimodal function $f1$ is relatively prone to be influenced by the control parameter settings. From Table I, it is also seen that all P-values of the interaction between p_l and p_e are greater than the 5% significance level. Namely, the interactive effect of the two key control parameters is statistically significant. The interactive effect is mainly derived from the p_e parameter because P-values with respect to p_e are more than P-values with respect to p_l, apart from function $f1$.

In the proposed turbulent operator, the over-layer moving operation arbitrarily searches a candidate solution in a limited range containing $(m-1)*(n-1)$ neighbor solutions. If the value of p_e is too small, the chance of the over-layer moving operation increases, and subsequently the risk of premature convergence rises. This is why the p_e parameter is relatively prone to impact on the performance of the WFO algorithm.

TABLE I
P-Values of the Multiway Analysis of Variance for Testing the Effect of Parameters

Source	f1	f5	f14	f24
m	0.075	0	0	0
p_l	0.116	0	0	0
p_e	0.068	0.027	0.025	0.206
$m * p_l$	0.002	0	0	0
$m * p_e$	0.440	0	0	0.386
$p_l * p_e$	0.617	0.076	0.141	0.667

TABLE II
Parameter Settings for Competitive Algorithms

Algorithm	Parameters
PSO	$PS = 50, \omega = 0.7298, c_1 = c_2 = 1.4961.$
DE	$PS = 50, F = 0.5, CR = 0.1.$
HS	$HMS = 10, HMCR = 0.8, PAR = 0.3, Bw = 0.001.$
ABC	$SN = 50, limit = 100.$
GSA	$PS = 50, G_0 = 100, \alpha = 20, K_0 = 50.$
WWO	$PS = 50, h_{\max} = 6, k_{\max} = \min(12, \lceil 0.5n \rceil), \alpha = 1.026, \beta_{\max} = 0.25, \beta_{\min} = 0.001.$
GWO	$PS = 50.$
SSA	$PS = 50, r_a = 1, p_c = 0.7, p_m = 0.1.$
FA	$PS = 50, \alpha_0 = 0.5, \alpha(t+1) = \alpha(t) * (\frac{10^{-4}}{0.9})^{\frac{1}{nfe}}, \beta = 0.5\exp^{-\gamma * r^2} + 0.5, \gamma = 1.$
jSO	$PS = \lceil 25\log(n)\sqrt{n} \rceil, H = 5, M_{CR} = 0.5; M_F = 0.5; p_{\min} = 0.125, p_{\max} = 0.25.$
WFO	$m = 50, p_l = 0.3, p_e = 0.7.$

C. Comparison With Related Metaheuristics

The WFO algorithm was compared with nine metaheuristics: 1) PSO [41], DE [36], HS [27], ABC [11], GSA [23],

TABLE III
MEAN AND STANDARD DEVIATION (STD) OF ABSOLUTE ERRORS ON THREE UNIMODAL AND SEVEN SIMPLE MULTIMODAL FUNCTIONS WITH **10** DIMENSIONS.

Fun.	Sta.	PSO	DE	HS	ABC	GSA	WWO	GWO	SSA	FA	jSO	WFO
f1	mean	3.82E+08	2.60E+02	8.64E+03	2.30E+02	1.92E+03	1.16E+03	1.36E+07	4.76E+02	1.37E+03	**0**	1.56E-13
	std	7.60E+08	3.76E+02	1.83E+04	1.92E+02	2.12E+03	1.56E+03	6.29E+07	7.50E+02	1.31E+03	0	1.45E-13
f2	mean	9.69E+07	2.64E-05	5.04E+02	5.76E-04	5.41E+10	2.20E+03	4.12E+06	1.72E+03	9.92E+01	**0**	1.99E-14
	std	4.21E+08	2.44E-05	9.88E+02	6.60E-04	7.01E+10	3.90E+03	1.21E+07	5.14E+03	1.93E+02	0	1.69E-14
f3	mean	9.47E-15	9.25E+02	2.94E+02	7.89E+03	1.35E+04	1.43E+01	1.16E+03	1.24E+03	2.90E-04	**0**	1.02E-13
	std	2.15E-14	5.02E+02	2.55E+02	3.58E+03	4.75E+03	4.71E+01	2.17E+03	7.17E+02	9.97E-05	0	1.07E-13
f4	mean	2.50E+01	5.08E+00	8.17E+00	1.90E-01	1.13E+01	5.27E+00	1.09E+01	1.01E+00	4.42E+00	**0**	7.96E-14
	std	4.09E+01	1.40E+00	1.33E+01	1.29E-01	6.28E+00	1.74E+00	1.17E+01	8.31E-01	3.18E-01	0	9.63E-14
f5	mean	2.07E+01	4.42E+00	7.93E+00	7.42E+00	3.51E+01	1.59E+01	1.46E+01	6.38E+00	5.04E+00	**2.59E+00**	5.19E+00
	std	7.28E+00	1.21E+00	2.76E+00	2.05E+00	1.48E+01	7.03E+00	7.60E+00	1.93E+00	2.48E+00	1.09E+00	1.59E+00
f6	mean	1.16E+00	4.17E-14	1.34E-01	4.34E-10	1.20E+00	3.17E+00	3.85E-01	6.22E-04	1.53E-02	**0**	5.78E-07
	std	1.83E+00	5.57E-14	4.35E-02	1.56E-09	2.38E+00	2.42E+00	6.50E-01	3.41E-04	9.13E-03	0	6.73E-07
f7	mean	2.00E+01	1.57E+01	2.46E+01	1.74E+01	3.36E+01	3.50E+01	2.64E+01	1.63E+01	1.63E+01	**1.33E+01**	1.40E+01
	std	5.49E+00	1.49E+00	6.06E+00	2.05E+00	7.54E+00	1.35E+01	7.90E+00	1.87E+00	3.53E+00	9.94E-01	4.02E+00
f8	mean	1.55E+01	4.74E+00	9.63E+00	8.05E+00	1.94E+01	1.84E+01	1.27E+01	6.69E+00	5.87E+00	**2.65E+00**	6.18E+00
	std	7.79E+00	1.31E+00	3.48E+00	2.08E+00	7.73E+00	8.78E+00	5.08E+00	1.66E+00	2.83E+00	1.16E+00	1.96E+00
f9	mean	1.95E+01	**0**	2.29E+00	6.34E-02	1.41E-01	2.71E-02	4.51E+00	1.79E-01	4.28E-05	**0**	7.20E-14
	std	7.99E+01	0	2.98E+00	1.53E-01	1.93E-01	8.64E-02	1.14E+01	2.15E-01	1.46E-05	0	6.32E-14
f10	mean	6.51E+02	2.80E+02	**2.08E+02**	2.76E+02	1.37E+03	8.17E+02	5.75E+02	2.75E+02	2.88E+02	2.22E+02	2.25E+02
	std	2.59E+02	8.70E+01	1.28E+02	1.01E+02	3.72E+02	2.72E+02	3.09E+02	9.66E+01	2.61E+02	1.40E+02	1.43E+02
Sr		89	36	68	52	100	82	87	57	48	12	29
Rank		10	3	7	5	11	8	9	6	4	1	2

Sr: summation of algorithm ranks in ascending order of mean error for each test function. Rank: rank in ascending order of Sr.

TABLE IV
MEAN AND STANDARD DEVIATION (STD) OF ABSOLUTE ERRORS ON THREE UNIMODAL AND SEVEN SIMPLE MULTIMODAL FUNCTIONS WITH **50** DIMENSIONS

Fun.	Sta.	PSO	DE	HS	ABC	GSA	WWO	GWO	SSA	FA	jSO	WFO
f1	mean	2.62E+10	3.97E+03	4.28E+09	3.54E+03	7.81E+08	1.04E+09	6.79E+09	6.93E+02	1.16E+04	**3.93E-14**	4.90E-12
	std	1.19E+10	2.59E+03	5.74E+08	2.04E+03	1.18E+09	4.51E+08	2.91E+09	5.52E+02	8.67E+03	1.78E-14	1.65E-11
f2	mean	4.31E+73	2.04E+42	2.21E+53	8.89E+23	2.54E+66	4.06E+56	1.44E+53	2.42E+79	2.69E+06	**5.56E-11**	9.10E+01
	std	2.33E+74	3.82E+42	5.02E+53	4.54E+24	1.04E+67	1.37E+57	7.89E+53	9.83E+79	1.35E+07	2.92E-10	1.76E+02
f3	mean	3.84E+04	1.30E+05	6.19E+04	2.16E+05	1.94E+05	1.26E+05	8.01E+04	8.74E+04	3.32E+03	**6.54E-13**	3.03E+03
	std	4.43E+04	1.23E+04	9.06E+03	2.12E+04	1.37E+04	2.21E+04	1.69E+04	1.26E+04	2.79E+03	2.21E-13	2.97E+03
f4	mean	3.87E+03	1.35E+02	5.92E+02	3.69E+01	2.40E+02	6.49E+02	5.98E+02	7.85E+01	1.38E+02	4.47E+01	**2.84E+01**
	std	2.72E+03	1.42E+01	8.21E+01	1.52E+01	1.23E+02	1.81E+02	2.62E+02	2.95E+01	5.43E+01	3.53E+01	4.23E+01
f5	mean	3.08E+02	2.24E+02	3.47E+02	2.04E+02	2.65E+02	1.99E+02	1.96E+02	1.75E+02	5.68E+01	**1.49E+01**	1.46E+02
	std	7.35E+01	1.39E+01	3.32E+01	2.16E+01	2.78E+01	3.73E+01	2.92E+01	1.40E+01	1.70E+01	6.44E+00	2.25E+01
f6	mean	2.99E+01	**3.26E-13**	1.00E+01	9.96E-12	4.03E+00	3.61E+01	1.31E+01	1.85E-10	7.90E-02	6.77E-07	3.91E-05
	std	7.80E+00	3.93E-14	1.20E+00	1.41E-11	4.79E+00	8.64E+00	4.16E+00	1.84E-11	1.24E-02	8.04E-07	9.45E-05
f7	mean	7.09E+02	2.77E+02	5.75E+02	2.33E+02	2.02E+02	2.69E+02	3.47E+02	2.29E+02	**1.16E+02**	1.83E+02	1.77E+02
	std	2.35E+02	1.23E+01	3.10E+01	1.99E+01	3.67E+01	6.09E+01	6.50E+01	1.42E+01	1.99E+01	2.47E+01	2.32E+01
f8	mean	3.31E+02	2.19E+02	3.44E+02	2.12E+02	2.62E+02	2.14E+02	1.95E+02	1.74E+02	5.27E+01	**2.43E+01**	1.38E+02
	std	5.26E+01	1.22E+01	3.48E+01	1.77E+01	2.53E+01	6.26E+01	3.22E+01	1.90E+01	1.13E+01	2.77E+01	2.57E+01
f9	mean	9.90E+03	7.77E+01	2.24E+03	8.43E+03	3.06E+04	2.82E+03	5.53E+03	7.97E+03	2.60E-02	**1.02E-13**	5.57E+02
	std	4.24E+03	5.50E+01	3.65E+02	2.00E+03	6.24E+03	1.34E+03	3.12E+03	2.35E+03	8.58E-02	3.47E-14	3.21E+02
f10	mean	7.11E+03	8.79E+03	1.18E+04	4.85E+03	6.20E+03	6.96E+03	5.59E+03	6.96E+03	**3.34E+03**	9.47E+03	3.99E+03
	std	9.69E+02	4.54E+02	7.94E+02	3.34E+02	6.63E+02	9.14E+02	6.92E+02	3.81E+02	9.32E+02	4.21E+02	4.21E+02
Sr		95	61	85	54	78	77	70	56	32	26	26
Rank		11	6	10	4	9	8	7	5	3	1	2

Sr: summation of algorithm ranks in ascending order of mean error for each test function. Rank: rank in ascending order of Sr.

FA [14], WWO [25], GWO [16], and SSA [21] and a self-adaptive variant of the state-of-the-art DE, jSO [42]. The former four algorithms are very popular in the soft computation community, whereas the other competing algorithms apart from the last one are representative of new metaheuristics over the latest decade. The jSO algorithm is the previous winner of the competition issued by the IEEE congress on evolutionary computation in 2017.

Each benchmark function in the open CEC2017 test suite was repeatedly solved by each competing algorithm over 30 times. For the sake of fairness, all comparison algorithms stopped after the maximal number of function evaluations, $nfe = 10\ 000 * n$, where n is the dimension of the problem to be solved. The parameter settings for all comparison algorithms are the same as those in their original articles, seeing Table II.

1) Comparison on Unimodal and Simple Multimodal Functions: Tables III and IV summarize the comparison results on three unimodal and seven simple multimodal functions. All competing algorithms are sorted in ascending order of the summation of their rank in each contrastive group. The final rank is listed at the bottom of every table. As shown in Tables III and IV, the WFO algorithm ranks second. However, both the average errors and standard deviations yielded by the WFO algorithm are close to those produced by the jSO algorithm and markedly less than those yielded by nine original metaheuristics regardless of ten or 50 dimensions, especially in solving the top four functions.

2) Comparison on Complicated Hybrid Functions: Tables V and VI, respectively, report the comparison results of all competing algorithms on ten hybrid functions with ten and 50 dimensions. The WFO algorithm as well as the jSO algorithm still achieve the best ranks because the average errors yielded by them are much less than those produced by the other nine metaheuristics in solving the majority of the complicated hybrid functions.

3) Comparison on Intractable Composition Functions: Tables VII and VIII, respectively, show the comparison results of all competing algorithms on 10 composition functions with 10 and 50 dimensions. From the two tables, we can observe that both the average errors and standard deviations yielded by the WFO algorithm are close to those produced by the ABC algorithm in solving the 10-D composition functions and those produced by algorithms jSO and FA in solving the 50-D composition functions. For a few functions, the average errors yielded by the WFO algorithm are the minimal among the same contrastive group, for example, the last benchmark function with dimensions 10 or 50.

TABLE V
MEAN AND STANDARD DEVIATION (STD) OF ABSOLUTE ERRORS ON TEN HYBRID FUNCTIONS WITH TEN DIMENSIONS

Fun.	Sta.	PSO	DE	HS	ABC	GSA	WWO	GWO	SSA	FA	jSO	WFO
f11	mean	6.55E+01	2.13E+00	7.22E+00	6.94E+00	2.54E+03	3.34E+01	2.68E+01	3.84E+00	2.65E+01	**3.41E-13**	6.41E-01
	std	1.19E+02	7.08E-01	2.83E+00	3.66E+00	1.94E+03	1.71E+01	2.48E+01	1.46E+00	1.78E+01	1.25E-12	1.00E+00
f12	mean	1.52E+06	3.33E+04	4.46E+04	5.36E+04	4.13E+05	1.57E+05	4.64E+05	1.66E+04	4.63E+05	**8.66E+00**	9.33E+00
	std	4.76E+06	1.80E+04	4.02E+04	3.32E+04	2.08E+06	2.14E+05	7.30E+05	1.56E+04	1.04E+06	3.04E+01	2.48E+01
f13	mean	7.69E+03	9.67E+01	1.13E+04	7.02E+02	1.37E+04	6.54E+03	1.07E+04	5.13E+01	1.00E+04	4.67E+00	**3.30E+00**
	std	8.08E+03	9.82E+01	9.65E+03	5.08E+02	1.09E+04	6.22E+03	5.08E+03	5.85E+01	6.67E+03	1.40E+00	2.78E+00
f14	mean	1.34E+02	4.85E-01	1.31E+03	2.26E+02	4.00E+03	1.46E+02	4.94E+02	4.64E+01	7.05E+01	**6.69E-02**	7.66E-01
	std	3.72E+02	4.58E-01	1.69E+03	2.87E+02	3.21E+03	1.87E+02	1.14E+03	4.29E+01	2.30E+01	2.52E-01	6.55E-01
f15	mean	2.79E+02	3.40E-01	2.85E+03	1.20E+02	1.63E+04	7.75E+01	1.38E+03	2.23E+01	4.41E+02	3.51E-01	**1.97E-01**
	std	9.26E+02	1.55E-01	3.58E+03	1.62E+02	1.36E+04	6.77E+01	1.26E+03	1.78E+01	3.96E+02	2.03E-01	2.74E-01
f16	mean	1.12E+02	2.54E+00	5.26E+01	4.21E+00	3.64E+02	9.84E+01	9.41E+01	2.23E+00	4.66E+01	1.23E+00	**8.04E-01**
	std	1.08E+02	2.58E+00	6.70E+01	5.10E+00	1.66E+02	1.00E+02	1.02E+02	2.12E+00	5.47E+01	2.04E+00	4.14E-01
f17	mean	6.57E+01	**1.72E-01**	1.27E+01	2.34E+00	1.49E+02	5.23E+01	4.61E+01	3.12E+00	3.46E+01	1.08E+01	1.35E+00
	std	4.83E+01	1.78E-01	1.35E+01	1.58E+00	8.26E+01	2.45E+01	1.40E+01	1.79E+00	1.58E+01	7.62E+00	5.38E-01
f18	mean	1.56E+04	8.26E-01	4.88E+03	1.49E+03	2.08E+04	8.45E+03	2.94E+04	3.48E+02	8.53E+03	**2.01E-01**	2.64E-01
	std	1.24E+04	3.63E+00	4.94E+03	1.06E+03	1.47E+04	7.98E+03	1.44E+04	3.54E+02	6.87E+03	2.04E-01	3.32E-01
f19	mean	1.37E+03	**2.91E-03**	2.83E+03	1.02E+02	1.24E+04	4.68E+01	3.94E+03	1.27E+01	3.79E+02	7.54E-03	5.69E-02
	std	5.66E+03	4.74E-03	4.02E+03	1.20E+02	1.12E+04	3.96E+01	5.42E+03	1.70E+01	9.67E+02	1.01E-02	3.92E-02
f20	mean	4.02E+01	**1.04E-02**	4.34E+00	6.35E-01	2.43E+02	6.13E+01	6.26E+01	5.79E-01	3.14E+01	1.50E+00	2.71E-01
	std	4.34E+01	5.70E-02	5.38E+00	5.30E-01	7.81E+01	3.96E+01	5.29E+01	4.35E-01	1.02E+01	5.09E+00	2.99E-01
Sr		86	25	75	53	106	73	92	36	72	23	19
Rank		9	3	8	5	11	7	10	4	6	2	1

Sr: summation of algorithm ranks in ascending order of mean error for each test function. Rank: rank in ascending order of Sr.

TABLE VI
MEAN AND STANDARD DEVIATION (STD) OF ABSOLUTE ERRORS ON TEN HYBRID FUNCTIONS WITH 50 DIMENSIONS

Fun.	Sta.	PSO	DE	HS	ABC	GSA	WWO	GWO	SSA	FA	jSO	WFO
f11	mean	1.05E+03	1.22E+02	8.07E+02	2.02E+03	2.76E+04	2.36E+02	2.45E+03	7.99E+01	2.56E+02	**3.00E+01**	8.44E+01
	std	1.00E+03	1.71E+01	3.36E+02	7.39E+02	4.06E+03	5.55E+01	1.66E+03	1.01E+01	4.15E+01	3.71E+00	2.68E+01
f12	mean	6.86E+09	1.16E+07	1.38E+08	5.52E+06	5.06E+08	7.75E+07	6.43E+08	3.59E+06	7.03E+06	**1.50E+03**	3.57E+04
	std	4.93E+09	3.52E+06	5.26E+07	1.57E+06	6.97E+08	4.58E+07	7.35E+08	1.51E+06	3.70E+06	3.30E+02	3.10E+04
f13	mean	3.16E+09	2.16E+04	7.64E+04	2.10E+04	2.35E+08	1.76E+05	1.26E+08	4.37E+03	1.62E+05	**3.52E+01**	1.06E+02
	std	3.48E+09	1.05E+04	3.44E+04	1.33E+04	6.56E+08	2.37E+05	1.17E+08	1.36E+03	1.28E+05	1.78E+01	2.50E+01
f14	mean	1.27E+06	7.54E+05	1.63E+06	9.08E+05	1.04E+06	1.04E+05	9.70E+05	1.75E+05	3.01E+04	**2.57E+01**	9.41E+01
	std	2.81E+06	3.91E+05	1.33E+06	2.89E+05	2.38E+06	7.36E+04	1.03E+06	7.64E+04	2.35E+04	2.64E+00	2.36E+01
f15	mean	6.61E+07	8.51E+03	6.96E+03	1.86E+04	4.40E+04	8.22E+03	1.07E+07	9.08E+02	5.09E+04	**2.52E+01**	5.70E+01
	std	1.95E+08	3.47E+03	5.61E+03	1.95E+03	2.86E+04	4.90E+03	2.23E+07	3.18E+02	3.28E+04	3.09E+00	1.70E+01
f16	mean	2.42E+03	1.24E+03	3.02E+03	1.20E+03	1.62E+03	1.71E+03	1.32E+03	1.18E+03	5.74E+02	**9.06E+02**	1.03E+03
	std	6.23E+02	1.64E+02	3.99E+02	1.54E+02	4.25E+02	4.62E+02	3.24E+02	2.19E+02	1.82E+02	1.53E+02	3.21E+02
f17	mean	2.04E+03	8.48E+02	1.27E+03	9.44E+02	1.72E+03	1.36E+03	1.03E+03	7.31E+02	7.17E+02	**5.19E+02**	7.67E+02
	std	3.57E+02	1.51E+02	4.14E+02	1.24E+02	3.88E+02	2.78E+02	2.57E+02	1.35E+02	2.49E+02	1.29E+02	1.96E+02
f18	mean	9.37E+06	1.70E+06	5.35E+06	1.67E+06	3.43E+06	2.13E+06	3.17E+06	7.09E+05	2.76E+05	**2.56E+01**	7.22E+01
	std	2.36E+07	9.73E+05	3.45E+06	6.96E+05	8.43E+06	1.83E+06	3.82E+06	3.21E+05	2.22E+05	2.34E+00	2.29E+01
f19	mean	7.21E+07	8.13E+03	1.15E+04	2.59E+04	6.81E+04	1.83E+04	1.08E+07	6.42E+02	4.84E+05	**1.58E+01**	3.70E+01
	std	2.34E+08	3.71E+03	1.13E+04	7.30E+03	9.96E+04	1.19E+04	4.70E+07	6.91E+02	2.18E+05	3.40E+00	9.33E+00
f20	mean	1.12E+03	6.21E+02	8.03E+02	8.25E+02	1.35E+03	1.14E+03	7.48E+02	6.19E+02	**3.84E+02**	4.51E+02	5.42E+02
	std	2.73E+02	1.68E+02	3.19E+02	1.09E+02	3.90E+02	2.76E+02	1.87E+02	1.25E+02	1.80E+02	1.01E+02	1.85E+02
Sr		103	53	77	62	93	70	85	34	46	12	25
Rank		11	5	8	6	10	7	9	3	4	1	2

Sr: summation of algorithm ranks in ascending order of mean error for each test function. Rank: rank in ascending order of Sr.

TABLE VII
MEAN AND STANDARD DEVIATION (STD) OF ABSOLUTE ERRORS ON TEN COMPOSITION FUNCTIONS WITH TEN DIMENSIONS

Fun.	Sta.	PSO	DE	HS	ABC	GSA	WWO	GWO	SSA	FA	jSO	WFO
f21	mean	2.07E+02	1.43E+02	2.11E+02	1.13E+02	2.37E+02	1.54E+02	2.11E+02	**1.10E+02**	1.93E+02	1.24E+02	1.15E+02
	std	4.27E+01	3.54E+01	2.09E+01	3.92E+00	1.37E+01	5.88E+01	2.13E+01	2.70E+01	3.72E+01	4.47E+01	3.77E+01
f22	mean	1.14E+02	8.39E+01	9.86E+01	**5.40E+01**	1.20E+02	1.03E+02	1.14E+02	7.28E+01	1.00E+02	9.67E+01	5.88E+01
	std	2.39E+01	1.79E+01	1.78E+01	2.16E+01	2.38E+01	9.22E+00	4.27E+01	3.10E+01	1.06E+00	1.83E+01	4.85E+01
f23	mean	3.32E+02	3.08E+02	3.16E+02	2.93E+02	3.59E+02	3.19E+02	3.19E+02	**3.01E+02**	3.07E+02	3.03E+02	3.07E+02
	std	1.33E+01	1.73E+00	4.80E+00	7.20E+01	1.78E+01	9.84E+00	7.81E+00	5.46E+01	4.20E+00	1.74E+00	2.15E+00
f24	mean	3.38E+02	2.63E+02	3.45E+02	**9.72E+01**	3.60E+02	3.11E+02	3.41E+02	1.24E+02	3.26E+02	3.00E+02	1.11E+02
	std	8.88E+01	6.53E+01	6.41E+00	1.29E+01	8.24E+01	8.45E+01	1.38E+01	3.68E+01	4.27E+01	7.99E+01	6.49E+01
f25	mean	4.64E+02	4.03E+02	4.41E+02	**1.80E+02**	4.40E+02	4.35E+02	4.37E+02	2.85E+02	4.38E+02	4.07E+02	3.61E+02
	std	6.90E+01	7.87E+00	1.68E+01	7.21E+01	1.53E+01	2.06E+01	1.41E+01	9.19E+01	1.59E+01	1.85E+01	9.57E+01
f26	mean	5.60E+02	1.76E+02	5.18E+02	**6.96E+01**	4.59E+02	3.05E+02	5.14E+02	1.40E+02	3.00E+02	3.00E+02	2.20E+02
	std	3.32E+02	1.12E+02	3.99E+02	8.54E+01	3.68E+02	2.08E+01	3.88E+02	1.00E+02	2.82E-03	0.00E+00	8.87E+01
f27	mean	4.20E+02	3.90E+02	3.96E+02	3.94E+02	4.21E+02	3.97E+02	3.98E+02	3.94E+02	3.95E+02	**3.89E+02**	**3.88E+02**
	std	3.37E+01	5.93E-01	3.66E+00	2.35E+00	8.69E+00	3.36E+00	1.66E+01	2.11E+00	1.91E+00	1.37E-01	1.04E+00
f28	mean	6.00E+02	4.39E+02	4.06E+02	**2.42E+02**	5.69E+02	3.23E+02	5.36E+02	3.03E+02	5.63E+02	3.48E+02	2.93E+02
	std	1.56E+02	7.82E+01	6.27E+01	1.15E+02	6.76E+01	3.24E+01	1.08E+02	1.16E+02	1.06E+02	1.10E+02	9.98E+01
f29	mean	3.23E+02	2.54E+02	2.62E+02	**2.34E+02**	5.02E+02	3.02E+02	2.84E+02	2.63E+02	2.65E+02	2.37E+02	2.39E+02
	std	6.32E+01	9.36E+00	1.65E+01	3.84E+01	1.21E+02	4.50E+01	4.41E+01	1.40E+01	2.04E+01	2.73E+00	8.92E+00
f30	mean	4.40E+05	1.07E+04	1.28E+05	6.75E+03	5.05E+05	3.97E+05	5.59E+05	6.33E+03	3.63E+05	2.76E+04	**4.04E+02**
	std	7.84E+05	1.39E+04	1.95E+05	9.12E+03	9.12E+05	6.85E+05	6.89E+05	6.50E+03	7.15E+05	1.49E+05	2.25E+01
Sr		98	44	77	16	103	71	87	29	69	41	25
Rank		10	5	8	1	11	7	9	3	6	4	2

Sr: summation of algorithm ranks in ascending order of mean error for each test function. Rank: rank in ascending order of Sr.

4) Multiple Comparison Statistical Test: To confirm the relative superior performance of the proposed WFO algorithm on the open CEC2017 test suite, a multiple comparison nonparametric test was carried out based on the average error obtained by each competing algorithm in solving each benchmark function over 30 independent runs. The *P*-value of Friedman's test is equal to 2.73E-41 for 10-D benchmark functions and 1.46E-34 for 50-D ones. In the comparison experiment, the null hypothesis of the Friedman's test was that the average errors yielded by all competing algorithms are all the same. Because both the two *P*-values are much less than 0.001, we must reject the null hypothesis regardless of 10 or 50 dimensions. Namely, all differences between the performance of the competing algorithms on the CEC2017 test suite are statistically significant.

To further investigate the difference between the proposed algorithm and other competitors, the Bonferroni procedure was implemented. In the experiment, the null hypothesis of the

TABLE VIII
MEAN AND STANDARD DEVIATION (STD) OF ABSOLUTE ERRORS ON TEN COMPOSITION FUNCTIONS WITH 50 DIMENSIONS

Fun.	Sta.	PSO	DE	HS	ABC	GSA	WWO	GWO	SSA	FA	jSO	WFO
f21	mean	5.68E+02	4.36E+02	5.75E+02	3.83E+02	4.86E+02	4.03E+02	3.80E+02	3.75E+02	2.50E+02	**2.25E+02**	3.49E+02
	std	6.19E+01	1.17E+01	2.63E+01	7.67E+01	5.15E+01	5.95E+01	3.11E+01	1.85E+01	1.16E+01	2.33E+01	2.64E+01
f22	mean	7.18E+03	8.97E+03	1.26E+04	4.46E+03	7.65E+03	7.00E+03	5.87E+03	6.99E+03	**2.67E+03**	8.23E+03	4.51E+03
	std	9.52E+02	1.27E+03	5.65E+02	2.02E+03	1.49E+03	1.44E+03	1.05E+03	2.24E+03	1.51E+03	3.29E+03	9.41E+02
f23	mean	1.17E+03	6.60E+02	8.20E+02	6.47E+02	1.09E+03	7.06E+02	6.47E+02	6.10E+02	4.64E+02	**4.46E+02**	6.08E+02
	std	1.91E+02	1.43E+01	3.47E+01	5.38E+01	1.02E+02	9.44E+01	5.73E+01	2.13E+01	1.75E+01	1.37E+01	3.61E+01
f24	mean	1.30E+03	7.97E+02	9.28E+02	9.41E+02	9.12E+02	7.28E+02	7.87E+02	7.30E+02	5.40E+02	**5.12E+02**	8.20E+02
	std	1.55E+02	1.62E+01	1.96E+01	1.49E+02	7.21E+01	5.11E+01	1.23E+02	2.67E+01	1.50E+01	4.32E+00	7.71E+01
f25	mean	1.61E+03	5.26E+02	8.60E+02	5.12E+02	5.96E+02	9.57E+02	1.07E+03	5.51E+02	**4.83E+02**	**4.82E+02**	5.12E+02
	std	1.29E+03	1.20E+01	6.21E+01	1.95E+01	3.30E+01	1.14E+02	3.16E+02	1.49E+01	8.22E+00	4.01E+00	2.99E+01
f26	mean	8.21E+03	3.31E+03	4.94E+03	1.33E+03	1.83E+03	3.86E+03	3.53E+03	2.60E+03	1.50E+03	**1.20E+03**	2.60E+03
	std	1.72E+03	1.91E+02	4.26E+02	1.24E+03	2.01E+03	5.82E+02	3.87E+02	4.08E+02	1.47E+02	6.01E+01	8.45E+02
f27	mean	1.27E+03	5.59E+02	7.77E+02	6.43E+02	9.42E+02	1.04E+03	8.14E+02	6.06E+02	5.48E+02	**5.16E+02**	6.15E+02
	std	2.55E+02	1.56E+01	5.46E+01	2.54E+01	1.66E+02	8.71E+01	9.01E+01	2.62E+01	1.74E+01	1.95E+01	7.32E+01
f28	mean	3.91E+03	4.71E+02	8.14E+02	4.81E+02	7.68E+02	1.09E+03	1.24E+03	5.23E+02	4.63E+02	**4.60E+02**	4.70E+02
	std	2.12E+03	1.78E+01	6.61E+01	8.70E+00	3.04E+02	1.74E+02	2.72E+02	7.54E+00	8.40E+00	8.92E+00	1.88E+01
f29	mean	2.63E+03	7.97E+02	1.15E+03	1.07E+03	2.32E+03	2.42E+03	1.29E+03	7.84E+02	6.90E+02	**6.75E+02**	8.03E+02
	std	7.59E+02	9.11E+01	3.31E+02	1.23E+02	5.89E+02	4.44E+02	2.34E+02	9.37E+01	1.46E+02	7.01E+01	2.06E+02
f30	mean	1.38E+08	7.59E+05	4.18E+06	8.86E+05	1.82E+07	5.78E+07	8.66E+07	7.67E+05	1.98E+07	6.20E+05	**6.11E+05**
	std	3.11E+08	6.03E+04	1.36E+06	6.51E+04	5.39E+06	1.73E+07	3.77E+07	5.53E+04	2.89E+06	4.35E+04	5.07E+04
Sr		105	57	86	50	78	80	74	46	26	19	39
Rank		11	6	10	5	8	9	7	4	2	1	3

Sr: summation of algorithm ranks in ascending order of mean error for each test function. Rank: rank in ascending order of Sr.

TABLE IX
P-VALUES OF THE BONFERRONI POST-HOC TEST OF THE WFO ALGORITHM AGAINST OTHER COMPETING ALGORITHMS

Dimension	WFO v.s.									
	PSO	DE	HS	ABC	GSA	WWO	GWO	SSA	FA	jSO
10	3.82E-13	1	5.78E-07	1	2.22E-18	1.42E-07	3.18E-12	1	3.47E-04	1
50	6.18E-15	0.09	4.26E-08	0.17	3.33E-08	5.32E-06	3.46E-06	1	1	1

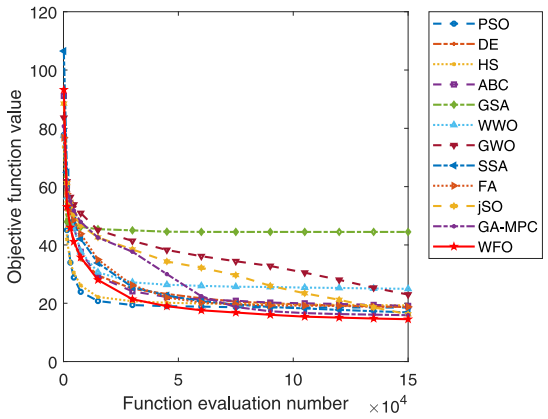

Fig. 4. Average convergence of all competing algorithms in solving the Messenger trajectory optimization problem over 30 independent runs.

Bonferroni *post-hoc* test is that the average errors yielded by the WFO algorithm are the same as those produced by other competitors. The *p*-values of the *post-hoc* test are listed in Table IX. From the table, it is clear seen that the *P*-values of the WFO algorithm against algorithms PSO, HS, GSA, WWO, and GWO are less than 0.001. Therefore, it is safe to make a statistical deduction that: 1) the WFO algorithm significantly outperforms algorithms PSO, HS, GSA, WWO, and GWO in solving the open CEC2017 test suite and 2) the performance differences between the WFO algorithm and algorithms SSA and jSO are not statistically significant.

VI. ENGINEERING APPLICATION IN SPACECRAFT TRAJECTORY OPTIMIZATION

The spacecraft trajectory optimization is a challenging optimization problem in the astronautical engineering domain. This problem aims at locating the best possible trajectory

TABLE X
RESULTS ON THE MESSENGER TRAJECTORY OPTIMIZATION PROBLEM OVER 30 INDEPENDENT RUNS

Algorithm	Best	Worst	Mean	Std.	Time	Rank
PSO	17.2449	29.0434	18.5241	4.5940	136.24	5
DE	11.3870	22.0998	18.6862	2.2031	135.69	6
HS	12.5083	27.8953	19.2726	3.3024	134.57	9
ABC	11.7960	24.8650	19.0876	3.2175	134.95	8
GSA	31.4010	64.9054	44.4655	7.0694	138.01	12
WWO	17.2089	39.5976	24.9471	5.6411	159.03	11
GWO	16.5980	30.8896	22.9734	3.7748	135.77	10
SSA	12.4381	19.8303	16.8679	1.9729	137.77	4
FA	15.4155	31.4581	18.7542	3.4804	138.89	7
jSO	13.6486	18.8542	16.2007	1.1373	138.04	3
GA-MPC	10.4871	19.0838	15.9008	1.8580	68.39	2
WFO	**8.5403**	18.7734	**14.5240**	2.6279	135.20	1

Rank: rank in ascending order of Mean.

that an interplanetary probe equipped with a chemical propulsion engine may take to travel from the Earth to other planet or asteroid. Its mathematical formulation is presented in [43].

Besides the aforementioned heuristics, the WFO algorithm was also compared with the GA with a multiparent crossover (GA-MPC) [44], which is the winner of the competition on testing evolutionary algorithms on real-world optimization problems in 2011. The original parameter setting for the GA-MPC algorithm was still adopted in this study. Unlike the GA-MPC algorithm, any parameters of other competing metaheuristics, including the proposed WFO algorithm were not meticulously tuned in order to accommodate the selected real-world optimization problems. The parameter settings for other competing algorithms completely keep the same as those listed in Table II.

All competing algorithms repeatedly solved two instantiations issued by the European Space Agency: 1) Messenger and 2) Cassini-2 [45]. Tables X and XI, respectively, present the experimental results regarding the two instances, which were obtained by all competing algorithms after 150 000 function evaluations. From the two tables, it is seen that the WFO algorithm achieves the top rank regardless of solving the 26-D Messenger instance or the 22-D Cassini-2 instance. As shown in Figs. 4 and 5, the convergent rate of the WFO algorithm is also quicker than those of other competitors on average, especially in solving the second instance.

TABLE XI
RESULTS ON THE CASSINI-2 TRAJECTORY OPTIMIZATION PROBLEM OVER 30 INDEPENDENT RUNS

Algorithm	Best	Worst	Mean	Std.	Time	Rank
PSO	15.4991	32.2938	24.5688	4.0023	124.82	10
DE	13.1539	22.2915	18.2956	1.9248	126.63	4
HS	9.9388	28.3063	21.9609	3.9220	124.85	7
ABC	13.4440	23.6804	19.0135	2.3797	125.90	6
GSA	39.0455	66.6232	50.6785	8.0925	128.71	12
WWO	19.8502	35.6548	26.0904	3.7230	146.61	11
GWO	14.3435	37.7712	22.1237	4.7932	126.59	8
SSA	13.5754	21.6009	18.3956	1.9395	128.58	5
FA	18.7085	32.6733	23.1009	3.2477	129.78	9
jSO	12.7553	21.2017	16.6073	2.7381	128.58	2
GA-MPC	**8.8483**	23.0057	17.8953	4.0675	63.87	3
WFO	9.7943	19.5802	**14.6256**	3.3991	126.01	1

Rank: rank in ascending order of Mean.

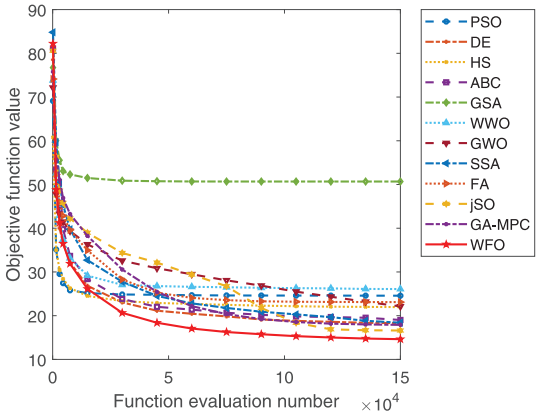

Fig. 5. Average convergence of all competing algorithms in solving the Cassini-2 trajectory optimization problem over 30 independent runs.

VII. CONCLUSION

In this article, a novel optimization algorithm called WFO was proposed. The WFO algorithm emulates the two types of water flow in hydraulics: 1) laminar and 2) turbulent. Based on the natural inspiration, the simulated model was well built from an optimization viewpoint. The convergence of the proposed optimizer was theoretically proved. The effectiveness and efficiency of the WFO algorithm were well examined through numerous comparison experiments with 11 related algorithms. The experimental results from the open CEC2017 test suite indicated that: 1) the WFO significantly outperforms algorithms PSO, HS, GSA, WWO, and GWO and 2) compared with algorithms DE, ABC, FA, SSA, and jSO, the WFO can achieve the competitive performance at the least computation consumption. The WFO is also successfully applied to solve two instantiations of spacecraft trajectory optimization. Moreover, the performance of the WFO algorithm on the selected real-world engineering optimization problem is the best in terms of solution quality and convergent rate.

The WFO algorithm has opened a new perspective toward improving other existing optimization algorithms. Both the regular search pattern and the integrated cooperation mechanism presented in the WFO algorithm are worthy of reference to improve other metaheuristics due to the effectiveness and efficacy of jumping over local optima and sharing good search information. In the future, the WFO algorithm can be improved by mimicking other natural phenomena of water flow, such as hydraulic drop and jump. In addition, it is also worthy of applications in various engineering problems.

REFERENCES

[1] K. Sorensen, M. Sevaux, and F. Glover, "A history of metaheuristics," in *Handbook of Heuristics*. Cham, Switzerland: Springer, 2018, pp. 1–18.
[2] H. Zang, S. Zhang, and K. Hapeshi, "A review of nature-inspired algorithms," *J. Bionic Eng.*, vol. 7, pp. S232–S237, Sep. 2010.
[3] D. Whitley, "A genetic algorithm tutorial," *Stat. Comput.*, vol. 4, no. 2, pp. 65–85, 1994.
[4] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization: An overview," *Swarm Intell.*, vol. 1, no. 1, pp. 33–57, 2007.
[5] D. P. Rini, S. M. Shamsuddin, and S. S. Yuhaniz, "Particle swarm optimization: Technique, system and challenges," *Int. J. Comput. Appl.*, vol. 14, no. 1, pp. 19–26, 2011.
[6] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization," *IEEE Trans. Cybern.*, vol. 45, no. 2, pp. 191–204, Feb. 2015.
[7] Y.-J. Gong *et al.*, "Genetic learning particle swarm optimization," *IEEE Trans. Cybern.*, vol. 46, no. 10, pp. 2277–2290, Oct. 2016.
[8] M. Dorigo and G. D. Caro, "Ant colony optimization: A new meta-heuristic," in *Proc. Congr. Evol. Comput.*, vol. 2, Jul. 1999, pp. 1470–1477.
[9] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theor. Comput. Sci.*, vol. 344, nos. 2–3, pp. 243–278, 2005.
[10] K. Socha and M. Dorigo, "Ant colony optimization for continuous domains," *Eur. J. Oper. Res.*, vol. 185, no. 3, pp. 1155–1173, 2008.
[11] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *J. Global Optimiz.*, vol. 39, no. 3, pp. 459–471, 2007.
[12] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, "A comprehensive survey: Artificial bee colony (ABC) algorithm and applications," *Artif. Intell. Rev.*, vol. 42, no. 1, pp. 21–57, 2014.
[13] K. Luo, "A hybrid binary artificial bee colony algorithm for the satellite photograph scheduling," *Eng. Optimiz.*, vol. 52, no. 8, pp. 1421–1440, 2020.
[14] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Proc. Int. Symp. Stochastic Algorithms*, 2009, pp. 169–178.
[15] I. Fister, Jr., X.-S. Yang, and J. Brest, "A comprehensive review of firefly algorithms," *Swarm Evol. Comput.*, vol. 13, pp. 34–46, Dec. 2013.
[16] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.
[17] K. Luo, "Enhanced grey wolf optimizer with a model for dynamically estimating the location of the prey," *Appl. Soft Comput.*, vol. 77, pp. 225–235, Apr. 2019.
[18] K. Luo and Q. Zhao, "A binary grey wolf optimizer for the multidimensional knapsack problem," *Appl. Soft Comput.*, vol. 83, Oct. 2019, Art. no. 105645.
[19] E. Cuevas, M. Cienfuegos, D. Zaldívar, and M. Pérez-Cisneros, "A swarm optimization algorithm inspired in the behavior of the social-spider," *Expert Syst. Appl.*, vol. 40, no. 16, pp. 6374–6384, 2013.
[20] E. Cuevas and M. Cienfuegos, "A new algorithm inspired in the behavior of the social-spider for constrained optimization," *Expert Syst. Appl.*, vol. 41, no. 2, pp. 412–425, 2014.
[21] J. J. Q. Yu and V. O. K. Li, "A social spider algorithm for global optimization," *Appl. Soft Comput.*, vol. 30, pp. 614–627, May 2015.
[22] D. Simon, "Biogeography-based optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 6, pp. 702–713, Dec. 2008.
[23] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A gravitational search algorithm," *Inf. Sci.*, vol. 179, no. 13, pp. 2232–2248, 2009.
[24] H. Shah-Hosseini, "The intelligent water drops algorithm: A nature-inspired swarm-based optimization algorithm," *Int. J. Bio-Inspired Comput.*, vol. 1, nos. 1–2, pp. 71–79, 2009.
[25] Y.-J. Zheng, "Water wave optimization: A new nature-inspired metaheuristic," *Comput. Oper. Res.*, vol. 55, pp. 1–11, Mar. 2015.
[26] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.
[27] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.
[28] K. Luo, J. Ma, and Q. Zhao, "Enhanced self-adaptive global-best harmony search without any extra statistic and external archive," *Inf. Sci.*, vol. 482, pp. 228–247, May 2019.
[29] T. Stutzle and M. Dorigo, "A short convergence proof for a class of ant colony optimization algorithms," *IEEE Trans. Evol. Comput.*, vol. 6, no. 4, pp. 358–365, Aug. 2002.

[30] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evol. Comput.*, vol. 10, no. 3, pp. 263–282, Sep. 2002.
[31] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf. Process. Lett.*, vol. 85, no. 6, pp. 317–325, 2003.
[32] M. Jiang, Y. P. Luo, and S. Y. Yang, "Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm," *Inf. Process. Lett.*, vol. 102, no. 1, pp. 8–16, 2007.
[33] H. Huang, C.-G. Wu, and Z.-F. Hao, "A pheromone-rate-based analysis on the convergence time of ACO algorithm," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 4, pp. 910–923, Aug. 2009.
[34] M. B. Abbott and A. W. Minns, *Computational Hydraulics*, 2nd ed. Abingdon, U.K.: Routledge, 2017.
[35] Y. Nakayama, *Introduction to Fluid Mechanics*. Oxford, U.K.: Butterworth-Heinemann, 2018.
[36] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optimiz.*, vol. 11, no. 4, pp. 341–359, 1997.
[37] W. Chu, X. Gao, and S. Sorooshian, "Handling boundary constraints for particle swarm optimization in high-dimensional search space," *Inf. Sci.*, vol. 181, no. 20, pp. 4569–4581, 2011.
[38] E. E. N. Mezura-Montes and C. A. C. Coello, "Constraint-handling in nature-inspired numerical optimization: Past, present and future," *Swarm Evol. Comput.*, vol. 1, no. 4, pp. 173–194, 2011.
[39] V. A. Zorich, "Limits," in *Mathematical Analysis I*. Berlin, Germany: Springer, 2015, pp. 79–149.
[40] N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization," Zhengzhou Univ., Zhengzhou, China, Rep. 201311, 2016.
[41] J. Kennedy, "Particle swarm optimization," *Encyclopedia Machine Learning*. Boston, MA, USA: Springer, 2010, pp. 760–766.
[42] J. Brest, M. S. Maučes, and B. Boškovic, "Single objective real-parameter optimization: Algorithm jSO," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2017, pp. 1311–1318.
[43] D. Izzo, V. M. Becerra, D. R. Myatt, S. J. Nasuto, and J. M. Bishop, "Search space pruning and global optimisation of multiple gravity assist spacecraft trajectories," *J. Global Optimiz.*, vol. 38, no. 2, pp. 283–296, 2007.
[44] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "GA with a new multi-parent crossover for constrained optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2011, pp. 857–864.
[45] S. Das and P. N. Suganthan, "Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems," Jadavpur Univ., Kolkata, India, Rep., 2010.

Kaiping Luo received the B.S. degree in applied mathematics from Sichuan Normal University, Chengdu, China, in 2004, the M.S. degree in applied mathematics from the University of Electronic Science and Technology of China, Chengdu, in 2007, and the Ph.D. degree in management science and engineering from the Harbin Institute of Technology, Harbin, China, in 2010.

He is currently an Associate Professor with the Department of Management Science and Engineering, Beihang University, Beijing, China. His research interests include evolutionary computation, operations research, and optimization.