Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 388 (2022) 114194
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications

Weiguo Zhao[a], Liying Wang[a,*], Seyedali Mirjalili[b,c]

[a] School of Water Conservancy and Hydropower, Hebei University of Engineering, Handan, Hebei, 056038, China
[b] Centre for Artificial Intelligence Research and Optimisation, Torrens University Australia, Fortitude Valley, Brisbane, 4006, QLD, Australia
[c] Yonsei Frontier Lab, Yonsei University, Seoul, Republic of Korea

Received 20 February 2021; received in revised form 12 September 2021; accepted 13 September 2021
Available online 9 November 2021

Abstract

A new bio-inspired optimization algorithm called artificial hummingbird algorithm (AHA) is proposed in this work to solve optimization problems. The AHA algorithm simulates the special flight skills and intelligent foraging strategies of hummingbirds in nature. Three kinds of flight skills utilized in foraging strategies, including axial, diagonal, and omnidirectional flights, are modeled. In addition, guided foraging, territorial foraging, and migrating foraging are implemented, and a visit table is constructed to model the memory function of hummingbirds for food sources. AHA is validated using two sets of numerical test functions, and the results are compared with those obtained from various algorithms. The comparisons demonstrate that AHA is more competitive than other meta-heuristic algorithms and determine high-quality solutions with fewer control parameters. Additionally, the performance of AHA is validated on ten challenging engineering design cases studies. The results show the superior effectiveness of AHA in terms of computational burden and solution precision compared with the existing optimization techniques in literature. The study also explores the application of AHA in hydropower operation design to further demonstrate its potential in practice. The source code of AHA is publicly available at https://seyedalimirjalili.com/aha and https://www.mathworks.com/matlabcentral/fileexchange/101133-artificial-hummingbird-algorithm?s_tid=srchtitle.

Keywords: Artificial hummingbird algorithm; Engineering optimization; Swarm intelligence; Meta-heuristics; Bio-inspired computing; Algorithm; Benchmark; Genetic algorithm

1. Introduction

During the past several decades, numerous optimization approaches have been designed to tackle a plurality of optimization problems. In recent years, however, the complexity of real-world optimization problems has emerged substantially with the development of human society and modern industry processes, which poses an even increasing challenge for optimization techniques. Generally speaking, the existing optimization techniques can be categorized into deterministic and meta-heuristic algorithms. Deterministic algorithms are specific mathematical functions and work mechanically and iteratively without any random nature. On a given problem, a deterministic method always obtains the same output for a particular input. Gradient descent and Newton's methods are two classic examples of deterministic algorithms. Although such algorithms can effectively find the local optima in solving some nonlinear

* Corresponding author.
E-mail addresses: zhaoweiguo@hebeu.edu.cn (W. Zhao), wangliying@hebeu.edu.cn (L. Wang), ali.mirjalili@gmail.com (S. Mirjalili).

https://doi.org/10.1016/j.cma.2021.114194
0045-7825/

problems, they may require derivative information of problems and tend to be trapped into the locally optimal solutions. Thus, these methods are powerless when dealing with highly constrained, complex problems with multiple peaks.

In view of the above, meta-heuristic methods have emerged to become ideal alternatives to deterministic methods. Such methods use different operators iteratively to explore and exploit the search space based on a minimum or maximum function [1]. These algorithms can balance between exploitation and exploration [2]. In the past two decades, meta-heuristic methods have been explored immensely and can be used in various fields in economy and trade, finance, energy, scheduling, image processing, optimal control, and engineering design applications [3–12]. Meta-heuristic methods are particularly popular due to their randomness and black box consideration of problems. Randomness makes meta-heuristics less sensitive to initial conditions and easier to switch between exploration and exploitation. The black box nature allows us to focus on the input and output rather than on the structural knowledge of considered problems. These merits enable meta-heuristics to effectively find global optimal solutions to given problems that deterministic methods cannot solve because of a lack of derivative or other related information.

Among meta-heuristic methods, bio-inspired algorithms have gained the most momentum in development in recent years and are increasingly applied to different engineering dilemmas with great success [13–15]. Bio-inspired algorithms typically mimic biological activities of living organisms and transform them into mathematical models in an optimized way. A great number of different living organisms exist in nature, some of which always offer us inspiration for developing effective optimization algorithms.

Genetic algorithm (GA), one of the most classic evolutionary algorithms (EAs), is originated from biological systems of natural selection to solve complex optimization problems [16]. The basic version of GA imitates three evolutionary behaviors: selection, crossover, and mutation. GA operates based on a population of individuals, each of which represents a candidate solution. The population evolves with these three operators over time, and the best individual so far is employed to generate a new population following numerous iterations. Due to the selection of individual proportional to their fitness values, GA converges to the global optimum. GA tends to be more powerful than deterministic methods and requires no additional information about the problems in question. This algorithm exhibits a good optimization capacity for solving complex problems, however, it readily suffers from premature convergence. Also, its optimization performance mostly depends on the election for the rates of crossover and mutation, the selection of objective functions, and population size [17].

Particle swarm optimization (PSO) is also one of the most popular bio-inspired methods [18] that imitates the social behaviors of bird flocking. The process starts with a population of random individuals whose positions are considered solutions to a problem. At each iteration, the position of each individual is stochastically updated based on the best global position obtained for any individual and the best position for the individual itself. The function fitness value is employed to measure the quality of an individual. Although PSO has a good convergence rate, it easily gets trapped into the local optima for some high-dimensional problems and is relatively sensitive to its control parameters [19].

Ant colony optimization (ACO), another popular bio-inspired algorithm, models the social behavior of ants when foraging [20]. When a population of ants explore the search space, they lay down pheromone trails to guide each other toward the targets. The goal of each ant is to search for the shortest path between the nest and the food source via pheromone trails. If any ant finds a shorter path, the rest of ants tend to follow this path until the positive feedback guides all ants to follow a single route. ACO is successfully applied to solve vehicle routing problems and other dynamic applications, but its theoretical analysis is difficult, and its convergence time is uncertain when solving problems [21].

Artificial bee colony (ABC), another well-regarded bio-inspired algorithm, models the intelligent foraging behaviors of honeybees [22]. In ABC, three groups of bees, including employed bees, onlooker bees, and scout bees, are used to perform global optimization. An employed bee performs local search for food based on visual information, and an onlooker bee chooses a food according to a probability proportional to the nectar amount. A scout bee randomly chooses a food in the search space. According to these three different foraging behaviors, ABC can balance exploration and exploitation and achieve fast convergence. Yet this algorithm may result in premature convergence in later iterations and the solution precision is sometimes unsatisfactory within an acceptable time limit [23].

Cuckoo search (CS) [24], an interesting bio-inspired algorithm, is based on the obligate brood parasitism of some cuckoo species. CS models two behaviors of cuckoos: reproduction and Levy flight. Cuckoos perform their

reproductive behaviors by laying eggs in the nest of host birds and then removing host birds' eggs to increase the hatching probabilities of their own eggs. Levy flight provides cuckoos with a special random walk characterized by the Levy distribution that combines a small-scale local search with occasional long-distance travel. This method is proved to be more efficient than PSO and GA in dealing with some complex optimization problems [25]. However, CS shows some disadvantages, such as low convergence precision and relatively poor local search ability [26,27].

Other successful bio-inspired meta-heuristics include, but not limited to, bat algorithm (BA) [28], glowworm swarm optimization (GSO) [29], fruit fly optimization algorithm (FOA) [30], bacterial foraging optimization (BFO) [31], squirrel search algorithm (SSA) [32], artificial ecosystem-based optimization [33], dolphin echolocation (DE) algorithm [34], shark smell optimization (SSO) [35], whale optimization algorithm (WOA) [36], virus colony search (VCS) [37], tree growth algorithm (TGA) [38], emperor penguin optimizer (EPO) [39], invasive weed optimization (IWO) [40], butterfly optimization algorithm (BOA) [41], spotted hyena optimizer (SHO) [42], krill herd (KH) [43], satin bowerbird optimizer (SBO) [44], runner-root algorithm (RRA) [45], bird mating optimization (BMA) [46], grasshopper optimization algorithm (GOA) [47], flower pollination algorithm (FPA) [48], crow search algorithm (CSA) [49], monkey king evolutionary (MKE) [50], and grey wolf optimizer (GWO) [51]. These meta-heuristics are based on the complex behaviors of living organisms to design different local and global search strategies, providing a larger selection of algorithms for scholars to tackle optimization problems in various fields.

It is noted that many bio-inspired meta-heuristics, such as ACO, ABC and PSO, are also population-based algorithms that simulate the collective behaviors of a social swarm on earth. However, some bio-inspired meta-heuristics only mimic behavior patterns and characteristics of individuals. For example, FPA performs local search by modeling biotic pollination, and performs a global search by modeling abiotic pollination [48,52].

It is worth mentioning that most bio-inspired meta-heuristics share common features, i.e., exploration and exploitation, irrespective of whatever they are inspired from. Exploration is a process of leaving any local region and subsequently exploring unknown spaces, while exploitation is a process of probing a local region to find a promising solution [53]. In general, exploration should be performed in early stages of algorithms, whereas exploitation is performed in later stages. A successful bio-inspired optimizer needs to first be equipped with effective mechanisms of exploratory and exploitative searches inspired from living things. Then, it requires to properly balance the trade-off between exploration and exploitation when searching for the global optimum for a given problem. The success of this model motivates us to develop an effective bio-inspired optimizer to tackle complex real-world problems.

One might be wondering why new bio-inspired optimizers are still being developed even though so many exist, and some of them perform well in solving complex problems. The answer can be found in No Free Lunch Theorem of Optimization [54], which theorizes that an algorithm that can effectively solve all kinds of optimization problems does not and will never exist. Therefore, this theorem encourages us to raise new, more efficient bio-inspired optimizers from different aspects, which motivates this study. Additionally, most optimizers have several control parameters. It is difficult to choose a set of different parameters well fitting different problems for a given optimizer. Hence, developing an algorithm with less control parameters is necessary, which is another motive of this study.

The algorithms mentioned above have been evaluated in the literature and the exploration and exploitation have been discovered in these algorithms. In GA [16], the exploration happens in the crossover and mutation phases, while the exploitation occurs in the selection phase. In ABC [22], the scout bees are employed to globally explore the search space, and the employed and onlooker bees are employed to locally explore the search space. In CS [25], the exploration is controlled by a Levy flight component and the exploitation is controlled by two individuals randomly chosen in the population. In BA [28], two parameters, the loudness and the pulse rate, are employed to control the exploration and exploitation, respectively. In AEO [33], the consumption factor encourages the algorithm to perform a global search and the decomposition factor assists the algorithm to perform a local search. In WOA [36], the shrinking encircling mechanism and spiral updating are used in the exploitation phase, while the searching for prey is done in the exploration phase. In VCS [37], the virus diffusion strategy contributes to the exploitation while the host cell infection strategy to the exploration. In BOA [41], the butterflies perform exploration by taking a step toward the fittest butterfly and the butterflies performs exploitation with respect to two butterflies which belong to the same swarm. In FPA [48], the flower pollens are carried by pollinators which enable flower pollens to move a long range, and this long-distance pollinator makes the exploration happen; different flowers of the same plant species make flower pollens move a short range, and this flower consistency makes exploitation occur. In GWO [51], the searching for prey of grey wolves improves the exploration ability of the algorithm, while the attacking prey of grey wolves enhances the exploitation ability of the algorithm. There are many existing algorithms with different

improved forms in the literature to strengthen the optimization ability of the algorithms. In the Gbest-guided ABC (GABC) [55], the search strategy is improved to enhance the exploitation, in which the information of the global optimum is incorporated into the solution search equation. In the enhanced WOA (EWOA) [56], the Levy flight strategy is introduced to improve the exploration and the ranking-based mutation operator is incorporated to enhance the exploitation. In the memory-based GWO (MGWO) [57], the search mechanism of the wolves is modified based on the information of each individual wolf, the crossover and greedy selection are introduced to improve the global and local search capacities.

AHA is quite different from the existing algorithms although it belongs to the category of meta-heuristics. The major difference between AHA and them is its particular biology background. AHA is inspired by three foraging strategies and three flight skills of hummingbirds in nature. Another important difference is the exploration and exploitation. In AHA, the migration foraging strategy guarantees exploration of the search space and the territorial foraging strategy promotes exploitation; meanwhile, the guided foraging strategy emphasizes exploration in the early stage and highlights exploitation in the later stage. The third variation between AHA and the existing algorithms is that AHA has a distinct memory update mechanism. Each hummingbird needs to know the last time to visit every other hummingbird, this information is recorded in a visit table, by which each hummingbird can choose its desired food source. Therefore, considering these mentioned factors, there are significant differences in AHA and the existing algorithms. There is a good work in literature that has tried to model the search behaviors of hummingbirds with Levy flight and the best individuals [58], but the proposed optimizer in this work is an attempt to mimic not only the search behaviors of hummingbirds with three foraging strategies, but also their superior memory and impressive flight skills.

This study introduces a new bio-inspired optimizer called artificial hummingbird algorithm (AHA). The inspiration for this algorithm is based on the special flight skills and intelligent foraging strategies of hummingbirds. AHA mimics three kinds of flight patterns involving axial, diagonal, and omnidirectional flights, along with three search strategies, guided foraging, territorial foraging, and migrating. In addition, an important component called visit table is introduced to implement the memory function of hummingbirds for seeking out and selecting food sources. The AHA algorithm is evaluated with two sets of various numerical test functions and a set of ten engineering cases and is compared with those of several other meta-heuristic techniques. Eventually, an example of hydropower operation design is considered in practice.

Section 2 describes the inspiration of AHA. The basic steps of the algorithm are introduced in detail in Section 3. Two sets of different numerical experiments and a set of ten engineering cases are utilized to investigate the performance of AHA in Section 4. Section 5 discusses the application of AHA in the hydropower operation design and Section 6 provides some concluding remarks and several directions for future study.

2. Inspiration

Hummingbirds are astounding animals and are considered to be the world's smallest birds. If their intelligence is measured by the brain-to-body ratio, hummingbirds would be the most intelligent animals on the earth, including humans [59]. About 360 species of hummingbirds are found worldwide, most species have a body length of only 7.5 to 13 cm.[1] Bee hummingbirds, the smallest hummingbirds, have an average length of 5.5 cm and a weight of 1.95 g.[2] Hummingbirds beat their wings with the highest frequency of all birds, up to 80 times per second. Generally, hummingbirds feed on various insects, such as mosquitoes, weevils, and aphids [60]. To supply enough energy for flying, hummingbirds also eat plenty of flower nectar and sweet liquid inside flowers every day. Fig. 1 shows a foraging hummingbird.[3]

What is special about hummingbirds is their amazing memory for foraging. A hummingbird has a hippocampus in its brain that plays a critical role in learning and memory, and is much larger than that of any other birds examined to date. Hummingbirds are tiny but extremely smart with a brain larger in relation to body size than any other birds, proving that hummingbirds have a prodigious memory [61]. In fact, each hummingbird can remember specific information about individual flowers in a certain region, including the location, nectar quality and contents [62], nectar-refilling rate [63], and the last time they visited the flowers. The birds also remember

[1] https://en.wikipedia.org/wiki/Hummingbird.

[2] https://en.wikipedia.org/wiki/Bee_hummingbird#cite_note-ADW-4.

[3] https://pixabay.com/zh/photos/hummingbird-bird-flowers-small-bird-5566297/.

Fig. 1. A foraging hummingbird.

spatial–temporal information about food sources. With this information in mind, hummingbirds can actually plan with some efficiency and avoid revisiting recently sampled flowers [61]. The employment and storage of memory about individual experiences are referred to as episodic memory, which was previously often used to distinguish animals from humans [64]. With this unique skill, hummingbirds become efficient foragers and tend to visit flowers that they have not visited for a long time for a more rewarding episode.

Another special skill is the flight ability of hummingbirds. The tiny body and high-frequency wingbeats make them the best flyers among bird species. With flexible shoulder joints, hummingbirds can rotate their wings 180 degrees and keep their wings moving in a figure-eight motion. This distinctive flight helps hummingbirds obtain strength from both the downstroke and upstroke [65], while other birds simply flap their wings to gain lift strength from the downstroke only. Hummingbirds can be considered the helicopters of the bird world since they often can be observed to rise like a helicopter. A hummingbird can fly in any direction with precision. Aside from the flight like other birds, hummingbirds can also fly at different attitudes, including forward and backward, up and down, and left and right [66]. Diagonal flying is also a peculiar flight posture that hummingbirds master like no other birds. They can fly in circles around a potential food resource when searching for food. Unbelievably, hummingbirds can stay in one spot in the air for a period of time. Hummingbirds are strongly migratory; they generally migrate to remote areas by flying thousands of miles on account of severe weather or food shortages.

The main inspirations behind AHA algorithm are the flight skills, memory capacity, and foraging strategies of hummingbirds. In the next section, these behaviors are mathematically modeled and an artificial hummingbird algorithm is proposed.

3. Artificial hummingbird algorithm (AHA)

A bio-inspired optimization algorithm, AHA, based on hummingbirds' intelligent behaviors, is presented in this section. The three main components of AHA are explained as follows.

Food sources: In reality, to select an appropriate source from a set of food sources, a hummingbird generally evaluates the properties of sources, including the nectar quality and content of individual flowers, nectar-refilling rate, and the last time to visit the flowers. For simplicity in AHA, it is assumed that each food source has the same number and identical type of flowers; a food source is a solution vector and the nectar-refilling rate of a food source is represented by function fitness value. The better the fitness value is, the higher the nectar-refilling rate of the food source will be.

Hummingbirds: Each hummingbird is always assigned to a specific food source from which it can be fed, then this hummingbird and the food source have the same position. A hummingbird can keep in mind the position and nectar-refilling rate of this specific food source and share the information with other hummingbirds in a population. In addition, for each hummingbird, it can also remember how long each food source is not visited by itself.

Visit table: The visit table records the visit level for each food source for different hummingbirds, which denotes the amount of time since the same hummingbird last visited a certain food source so far. The food source with a high visit level for a hummingbird will be given a priority visit for that hummingbird. To obtain more nectar, a

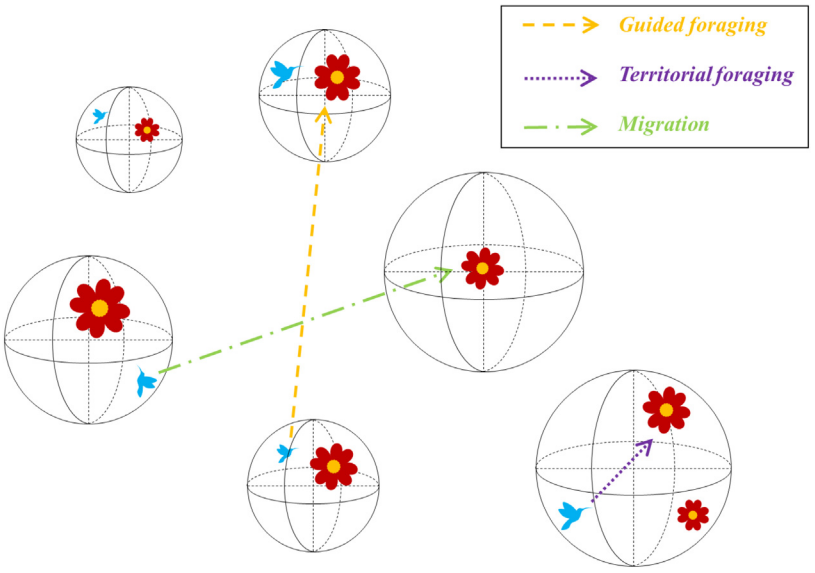

Fig. 2. Three foraging behaviors of AHA.

hummingbird tends to visit the food source with the highest nectar-refilling rate among the food sources with the same highest visit level. Each hummingbird is able to find its target food source via the visit table. The visit table is generally updated during each iteration.

The AHA algorithm is a swarm-based meta-heuristic method designed to solve optimization problems. This subsection provides three mathematical models simulating three foraging behaviors of hummingbirds: guided foraging, territorial foraging, and migrating foraging. These three foraging behaviors are depicted in Fig. 2. Similarly to most optimizers in swarm-based category, the structure of the proposed algorithm can be divided into three main stages. The general structure of AHA is provided in Algorithm 1.

Initialization
While stop criterion is not satisfied
    Guided foraging
    Territorial foraging
    Migration foraging
End

Algorithm 1. Structure of AHA.

3.1. Mathematical model and algorithm

3.1.1. Initialization

A population of n hummingbirds are placed on n food sources, randomly initialized as follows [67]:

$$x_i = Low + r \cdot (Up - Low) \qquad i = 1, \ldots, n \tag{1}$$

where Low and Up are the upper and lower boundaries for a d-dimensional problem, respectively, r is a random vector in $[0, 1]$, and x_i represents the position of the ith food source that is the solution of a given problem.

The visit table of food sources is initialized as follows:

$$VT_{i,j} = \begin{cases} 0 & if \quad i \neq j \\ \text{null} & i = j \end{cases} \quad i = 1, \ldots, n;\ j = 1, \ldots, n \tag{2}$$

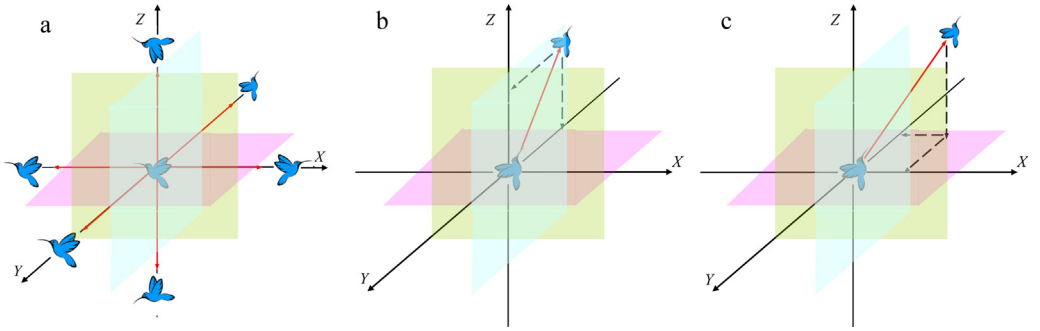

Fig. 3. Three flight behaviors of hummingbirds, (a) axial fight, (b) diagonal flight, (c) omnidirectional flight.

where for $i = j$, $VT_{i,j} = null$ indicates that a hummingbird is taking food at its specific food source; for $i \neq j$, $VT_{i,j} = 0$ indicates that the jth food source has just been visited by the ith hummingbird in the current iteration.

3.1.2. Guided foraging

There is a natural tendency for each hummingbird to visit the food source with the maximum nectar volume, which signifies that a target source needs to have a high nectar-refilling rate and a long unvisited time by that hummingbird. In AHA, therefore, a hummingbird is supposed to determine the food sources with the highest visit level for the guided foraging behavior, then it chooses the one with the highest nectar-refilling rate from them as its target food source. After the target food source is determined, this hummingbird can fly towards it for feeding.

During foraging, three flight skills, including omnidirectional, diagonal, and axial flights, are sufficiently used and modeled in the AHA algorithm by introducing a direction switch vector. This vector is used to control whether one or more directions in the d-dimension space are available. Fig. 3 shows the three flight behaviors in 3-D space. It can be seen that the axial flight shows a hummingbird can fly along any coordinate axis; the diagonal flight allows a hummingbird to move from one corner of a rectangle across to the opposite corner, and is determined by any two coordinate axes out of three. The omnidirectional flight shows that any flight direction can be projected to each of the three coordinate axes. In other words, all birds use omnidirectional flight, but only hummingbirds master the axial and diagonal flights.

These flight patterns can be extended to a d-D space, in which the axial flight is defined as follows:

$$D^{(i)} = \begin{cases} 1 & if \quad i = randi([1, d]) \\ 0 & else \end{cases} \qquad i = 1, \ldots, d \tag{3}$$

The diagonal flight is defined as follows:

$$D^{(i)} = \begin{cases} 1 & if \quad i{=}P(j),\ j \in [1, k],\ P = randperm(k),\ k \in [2, \lceil r_1 \cdot (d-2) \rceil + 1] \\ 0 & else \end{cases} \qquad i = 1, \ldots, d \tag{4}$$

The omnidirectional flight is defined as follows:

$$D^{(i)} = 1 \qquad i = 1, \ldots, d \tag{5}$$

where $randi([1, d])$ generates a random integer from 1 to d, $randperm(k)$ creates a random permutation of integers from 1 to k, and r_1 is a random number in (0, 1]. The diagonal flight in a d-D space is inside a hyperrectangle, which is bounded by any 2 to d-1 coordinate axes. The movement of a hummingbird in 3-D space using three flight skills is illustrated in Fig. 4, in which the red lines represent the omnidirectional flight, the green lines represent the diagonal flight, and the blue lines represent the axial flight. In the figure, a hummingbird is required to move from (4,4,4) to (0,0,0). After eight units of time, the hummingbird is capable of reaching the desired point by using three different flight skills. This shows that the mathematical models of the flight skills are able to mimic the searching behaviors of the hummingbirds in 3-D and multi-dimensional spaces.

With these flight abilities, a hummingbird visits its target food source, resulting in a candidate food source being obtained. As such, a food source is updated from the old one with respect to the target food source selected from

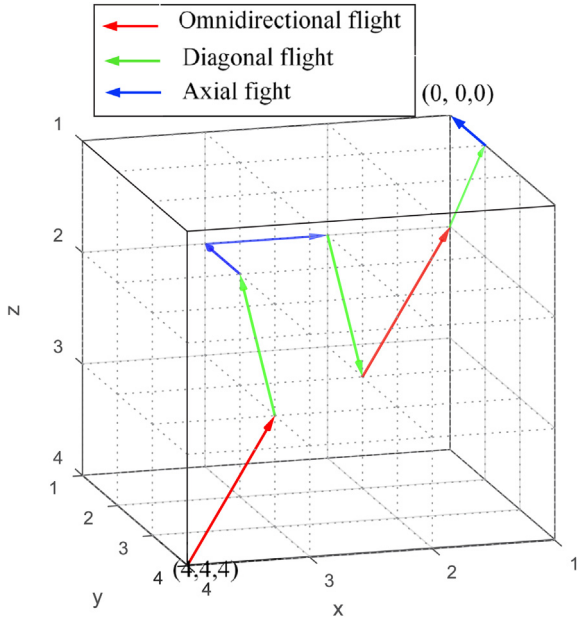

Fig. 4. Movement of a hummingbird in a 3-D space using three flight skills.

all the existing sources. The mathematical equation simulating the guided foraging behavior and a candidate food source is derived as follows:

$$v_i(t+1) = x_{i,tar}(t) + a \cdot D \cdot (x_i(t) - x_{i,tar}(t)) \tag{6}$$

$$a \sim N(0, 1) \tag{7}$$

where $x_i(t)$ is the position of the ith food source at time t, $x_{i,tar}(t)$ is the position of the target food source that the ith hummingbird intends to visit and a is a guided factor, which is subject to the normal distribution $N(0, 1)$ with mean = 0 and standard deviation = 1. Eq. (6) enables each current food source to update its position in the neighborhood of the target food source and models the guided foraging of hummingbirds via different flight patterns. The position update of the ith food source is as follows:

$$x_i(t+1) = \begin{cases} x_i(t) & f(x_i(t)) \le f(v_i(t+1)) \\ v_i(t+1) & f(x_i(t)) > f(v_i(t+1)) \end{cases} \tag{8}$$

where $f(\cdot)$ indicates the function fitness value. Eq. (8) shows that if the nectar-refilling rate of the candidate food source is better than that of the current one, the hummingbird abandons the current food source and stays at the candidate one resulted from Eq. (6) for feeding.

In the AHA algorithm, the visit table is an important component that stores the visit information of food sources. Any hummingbird can find its target food source that it wants to visit according to the visit table at each iteration. The visit table records how long each food source is not visited since last time it was visited by the same hummingbird, and a long unvisited time indicates a high visit level. Each hummingbird desires to choose the food source(s) with the highest visit level. If more than one source tie for the same highest visit level, the one with the best nectar-refilling rate is chosen as the target food source that a hummingbird will visit. Each hummingbird in the population visits its own target food source via Eq. (6). During each iteration, when a hummingbird performs the guided foraging using Eq. (6) with respect to its target food source, the visit levels of the other food sources for this hummingbird are incremented by 1 and the visit level of its target food source visited is initialized to 0. After performing the guided foraging, the hummingbird will not change its food source if there is not a better nectar-refilling rate (solution); the current source will be replaced by a new one if there is a better nectar-refilling rate (solution), and then this hummingbird will stay at the new food source. The update of a food source at which the corresponding hummingbird is resided indicates the visit level update of the food source for all other hummingbirds. The visit level to be updated is set as the highest level of the other food sources incremented by 1.

		Food source					
		x_1	x_2	x_3	x_4	x_5	x_6
Hummingbird	x_1	--	3	5	6	4	3
	x_2	2	--	4	6	**8**	5
	x_3	6	6	--	3	2	7
	x_4	5	4	6	--	8	2
	x_5	3	3	7	7	--	3
	x_6	6	2	5	6	3	--

Fig. 5. Visit table of a population of six hummingbirds.

Fig. 5 shows a visit table of a set of six food sources on which six hummingbirds are placed. The number in the visit table is the visit level signifying how long a hummingbird does not visit the food source. For example, the number '8' in blue indicates that the hummingbird x_2 does not visit the food source at which the hummingbird x_5 is resided for 8 time periods. The guided foraging strategy of AHA is shown in Algorithm 2.

For ith hummingbird from 1 to n
  Perform equation (6)
  If $f(v_i(t+1)) < f(x_i(t))$
    $x_i(t+1)=v_i(t+1)$
    For jth food source from 1 to n ($j \neq tar, i$)
      Visit_table(i,j)=Visit_table(i,j)+1
    End
    Visit_table(i,tar)=0
    For jth food source from 1 to n
      $\text{Visit\_table}(j,i) = \max_{l \in n \text{ and } l \neq j} (\text{Visit\_table}(j,l)) + 1$
    End
  Else
    For jth food source from 1 to n ($j \neq tar, i$)
      Visit_table(i,j)=Visit_table(i,j)+1
    End
    Visit_table(i,tar)=0
  End
End

Algorithm 2. Guided foraging strategy of AHA.

The following example (a minimization problem) shows how the visit table is maintained and how the target food source is chosen for each hummingbird in the guided foraging strategy.

Given a population of four hummingbirds, and their positions and the visit table are initialized using Eqs. (1) and (2), respectively. The first hummingbird finds three food sources that have the same highest visit level, among which the food source of the hummingbird x_4 has the highest nectar-refilling rate. Hence, this source is the target

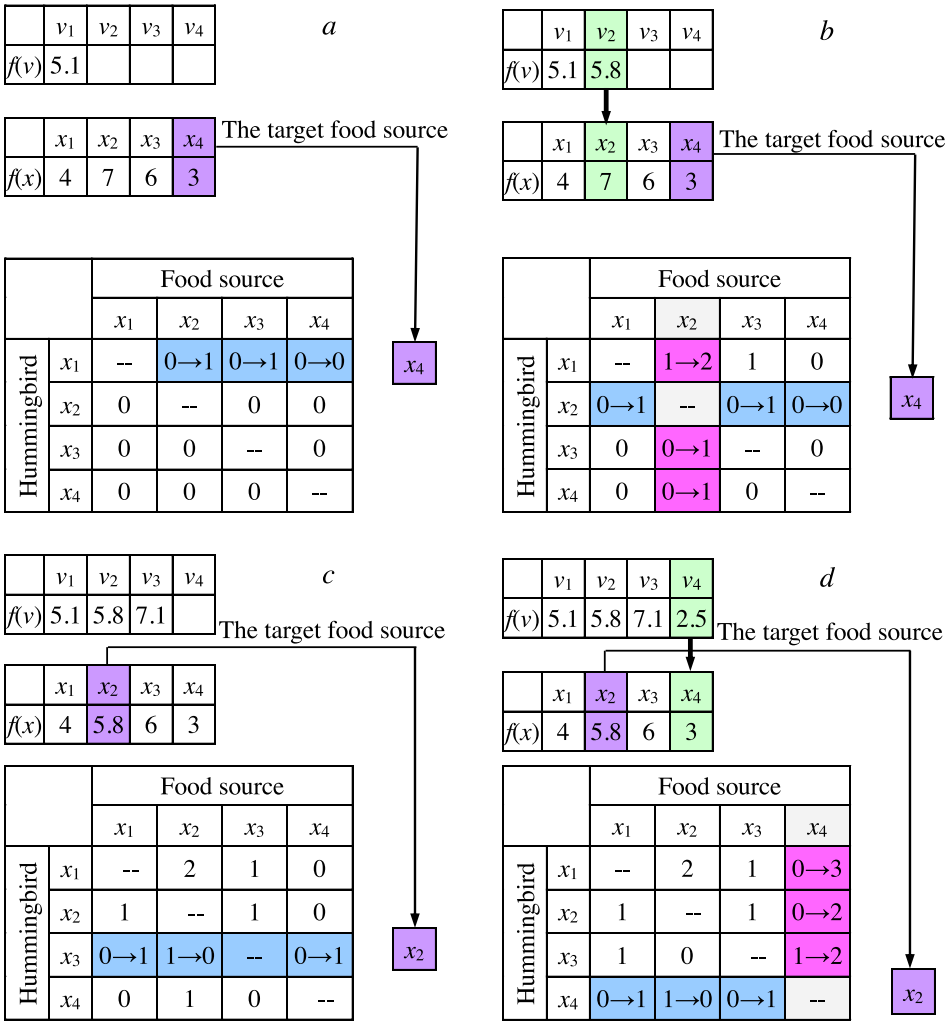

Fig. 6. Update of visit table and selection of the target food source during one iteration when performing guided foraging strategy.

food source of the first hummingbird. After Eqs. (6) and (8) are performed for this hummingbird, the visit levels of the food sources of the hummingbirds x_2 and x_3 are increased by 1 because neither are visited by the hummingbird x_1, and the target food source x_4 is initialized to 0. The update of the visit level and the selection of the target food source for the first hummingbird are shown in Fig. 6(a).

The second hummingbird finds three food sources with the same highest visit level, with the food source of the hummingbird x_4 having the highest nectar-refilling rate. Hence, the food source of the hummingbird x_4 is the target food source of the second hummingbird. After Eqs. (6) and (8) are performed for the second hummingbird, the visit levels of the sources of the hummingbirds x_1 and x_3 are increased by 1 and the target source x_4 is initialized to 0. The source of the hummingbird x_2 is replaced with the candidate v_2 since the nectar-refilling rate of the candidate v_2 is better than that of the source x_2, therefore, the visit level of the source x_2 for each of the other hummingbirds needs to be changed to the highest visit level increased by 1 in every corresponding row. The update of the visit level and the selection of the target food source for the second hummingbird are shown in Fig. 6(b).

For the third hummingbird, the food source of the hummingbird x_2 is the target food source owing to its highest visit level, so the visit levels of the food sources of the hummingbirds x_1 and x_4 are increased by 1, and the visit level of the target source x_2 is initialized to 0. The update of the visit level and the selection of the target food source for the third hummingbird are shown in Fig. 6(c).

For the fourth hummingbird, the food source of the hummingbird x_2 with the highest visit level is its target food source, so the visit level of the source x_2 is initialized to 0, and the visit levels of the sources of the hummingbirds

x_1 and x_3 are increased by 1. Since the source x_4 is replaced with its candidate v_4, the visit level of the source x_4 for each of the other hummingbirds needs to be changed to the highest visit level increased by 1 in every corresponding row. The update of the visit level and the selection of the target food source for the fourth hummingbird are shown in Fig. 6(d). After one iteration, the updated visit table for the hummingbirds is shown in Fig. 7.

3.1.3. Territorial foraging

After visiting its target food source where the flower nectar has been eaten, a hummingbird is likely to search for a new food source instead of visiting other existing food sources. Therefore, a hummingbird can readily move to its neighboring region within its own territory, in which a new food source may be found as a candidate solution that may be better than the current one. The mathematic equation simulating the local search of hummingbirds in the territorial foraging strategy and a candidate food source is obtained as follows:

$$v_i(t+1) = x_i(t) + b \cdot D \cdot x_i(t) \tag{9}$$

$$b \sim N(0, 1) \tag{10}$$

where b is a territorial factor, which is subject to the normal distribution $N(0,1)$ with mean = 0 and standard deviation = 1. Eq. (9) can allow any hummingbird to easily find a new food source in its local neighborhood according to its personal position by means of its special flight skills. After the territorial foraging strategy is performed, the visit table should be updated. The territorial foraging strategy of AHA is shown in Algorithm 3.

For *i*th hummingbird from 1 to *n*
  Perform equation (9)
  If $f(v_i(t+1)) < f(x_i(t))$
    $x_i(t+1) = v_i(t+1)$
    For *j*th food source from 1 to *n* ($j \neq i$)
      Visit_table(i,j)=Visit_table(i,j)+1
    End
    For *j*th food source from 1 to *n*
      $\text{Visit\_table}(j,i) = \max_{l \in n \text{ and } l \neq j} (\text{Visit\_table}(j,l)) + 1$
    End
  Else
    For *j*th food source from 1 to *n* ($j \neq i$)
      Visit_table(i,j)=Visit_table(i,j)+1
    End
  End
End

Algorithm 3. Territorial foraging strategy of AHA.

3.1.4. Migration foraging

When a region where a hummingbird frequently visits tends to be lack of food, this hummingbird usually migrates to a more distant food source for feeding. In the AHA algorithm, a migration coefficient is defined. If the number of iterations exceeds the predetermined value of the migration coefficient, the hummingbird locating at the food source with the worst nectar-refilling rate will migrate to a new food source produced randomly in the entire search space. At this time, this hummingbird will abandon the old source and stay at the new one for feeding, and then

	v_1	v_2	v_3	v_4
$f(v)$	5.1	5.8	7.1	2.5

	x_1	x_2	x_3	x_4
$f(x)$	4	5.8	6	2.5

		Food source			
		x_1	x_2	x_3	x_4
Hummingbird	x_1	--	2	1	3
	x_2	1	--	1	2
	x_3	1	0	--	2
	x_4	1	0	1	--

Fig. 7. Updated visit table of hummingbirds after one iteration.

the visit table is updated. The migration foraging of a hummingbird from the source with the worst nectar-refilling rate to a new one produced randomly can be given as follows:

$$x_{wor}(t+1) = Low + r \cdot (Up - Low) \tag{11}$$

where x_{wor} is the food source with the worst nectar-refilling rate in the population. The migration foraging strategy of AHA is shown in Algorithm 4.

If mod(t,2n)==0
  Perform equation (11)
    For jth food source from 1 to n ($j \neq wor$)
      Visit_table(wor,j)=Visit_table(wor,j)+1
    End
    For jth food source from 1 to n
      Visit_table(j,wor)= $\max_{l \in n \text{ and } l \neq j}$ (Visit_table(j,l)) +1
    End
End

Algorithm 4. Migration foraging strategy of AHA.

In the guided foraging strategy, when there is no food source whose position is adjusted, hummingbirds tend to move towards different sources as their respective target food sources, leading to a higher exploration and a lower probability of converging to the local optima. When there is a food source which is updated by a new one, this updated source more likely than the old one as the same target food source will guide hummingbirds stationed at the other different food sources to move towards it, resulting in a higher exploitation. Considering Eq. (6), in the early stage of iterations, exploration is emphasized on account of the long distance among food sources, while the distance adaptively decreases with the increase of iterations, thus exploitation is highlighted. In the territorial foraging strategy, a hummingbird carries out the exploitation process in its local neighborhood. Further, the migration foraging of hummingbirds indicates that a hummingbird has implemented the exploration process in the search space.

In AHA, in addition to two common parameters, the population size and maximum number of iterations, only one control parameter needs to be determined in considering whether the migration is performed. In the worst

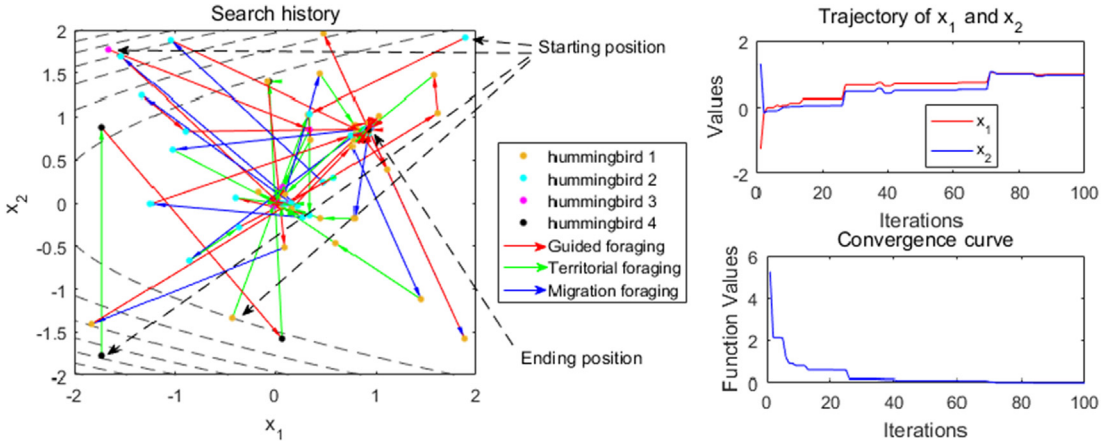

Fig. 8. Searching behaviors of AHA on Rosenbrock function.

case, if there is no replacement for all food sources when performing guided foraging and territorial foraging, a hummingbird will visit every food source in turn as its target source according to the visit table at each iteration. Assuming that there is a probability of 50% to choose between either the guided foraging or the territorial foraging, and there is the same probability of visiting every other source in the guided foraging. Thus, a hummingbird may visit the same food source as its target source after $2n$ iterations in the worst case. In this case, the migration foraging strategy needs to be performed to improve the stagnation and explore the search space. Therefore, the following definition for the migration coefficient relative to the population size is recommended as follows:

$$M = 2n \tag{12}$$

3.2. Pseudocode of AHA

AHA starts by initializing a set of random solutions and a visit table. At each iteration, there is a probability of 50% to perform either the guided foraging or the territorial foraging. The guided foraging allows hummingbirds to move toward their respective target food sources determined by the visit table and nectar-refilling rate. The territorial foraging forces hummingbirds to perturb their own local neighborhoods. With every $2n$ iterations, the migration foraging is performed. These three foraging behaviors all use the three flight skills, including omnidirectional, diagonal, and axial flights. All operations and calculations are carried out interactively until the stopping criterion is reached. Eventually, the food source with the best nectar-refilling rate is returned as an approximation of the global optimum. The pseudocode of AHA is given in Algorithm 5.

The computational complexity of the AHA algorithm is related to initialization, the fitness evaluation (c), the position update of hummingbirds, the size of the hummingbird population (N), the maximum number of iterations (T), and the dimension of variables (d). The overall computational complexity of AHA can be expressed as:

$$\begin{aligned} &O(AHA) = O(problem\ definition) + O(initialization) \\ &+O(t(function\ evaluation)) + O(t(guided\ foraging)) \\ &+O(t(territorial\ foraging)) + O(t(migration\ foraging)) \\ &= O(1 + nd + Tcn + \tfrac{1}{2}Tnd + \tfrac{1}{2}Tnd + \tfrac{T}{2n}nd) \\ &\cong O(Tcn + Tnd + \tfrac{Td}{2}) \end{aligned} \tag{13}$$

3.3. Searching behaviors

The searching behaviors of AHA are demonstrated using two classic test functions. The first is Rosenbrock function, which is a unimodal function with its global optimum falling in a long and narrow valley as depicted in Fig. 8. So, it is hard for an optimizer to find the optimal solution in a short time. The optimal solution is x = (1, 1) with $f(x)$=0. Fig. 8 shows the searching behaviors of this function, including search history, variable trajectories, and the convergence curve, meanwhile, the searching process of each of four hummingbirds for the global optimum is described clearly. The four hummingbirds start at different positions randomly produced in the search space, during

100 iterations, they can quickly approximate the optimum using three different flight skills. Obviously, AHA shows an excellent exploitation for the optimal solution and convergence rate. The algorithm arrives at a best solution x=(0.9724497, 0.9458298) with $f(x)$=7.6196E−4 using four hummingbirds after 100 iterations.

Input: n, d, f, *Max_Iteration*, *Low*, *Up*
Output: *Globalminimum*, *Globalminimizer*
Initialization:
  For ith hummingbird from 1 to n,
    Do x_i=*Low*+r(*Up*-*Low*),
    For jth food source from 1 to n, **Do**
      If $i\neq j$
      Then $Visit\_table_{i,j}$=1,
      Else $Visit\_table_{i,j}$=*null*,
      End If
    End For
  End For
While $t\leq$*Max_Iteration* **Do**
  For ith hummingbird from 1 to n, **Do**
    If *rand*≤0.5 **Then**
      If r<1/3 **Then** perform equation (3),
        Else If r>2/3
          Then perform equation (4),
          Else perform equation (5),
        End If
      End If
      Perform equation (6),
      If $f(v_i(t+1))<f(x_i(t))$
      Then $x_i(t+1)=v_i(t+1)$,
        For jth food source from 1 to n ($j\neq tar,i$), **Do**
        Visit_table(i,j) =Visit_table(i,j) +1,
        End For
        Visit_table(i,*tar*)=0,
        For jth food source from 1 to n, **Do**
Visit_table(j,i)= $\max_{l\in n \text{ and } l\neq j}$ (Visit_table(j,l)) +1,
      End For
    Else
      For jth food source from 1 to n ($j\neq tar,i$), **Do**
        Visit_table(i,j)=Visit_table(i,j)+1,
      End For
        Visit_table(i,*tar*)=0,
    End
  Else
    Perform equation (9),
    If $f(v_i(t+1))<f(x_i(t))$
    Then $x_i(t+1)=v_i(t+1)$,
      For jth food source from 1 to n ($j\neq i$), **Do**
      Visit_table(i,j)=Visit_table(i,j)+1,
      End For
      For jth food source from 1 to n, **Do**
      Visit_table(j,i) = $\max_{l\in n \text{ and } l\neq j}$ (Visit_table(j,l)) +1,
      End For
    Else
      For jth food source from 1 to n ($j\neq i$), **Do**
      Visit_table(i,j)=Visit_table(i,j)+1,
      End For
    End If
  End If
End For
If mod(t,2n)==0,
Then perform equation (11),
  For jth food source from 1 to n ($j\neq wor$), **Do**
  Visit_table(wor,j)=Visit_table(wor,j)+1,
  End For
  For jth food source from 1 to n, **Do**
  Visit_table(j,wor) = $\max_{l\in n \text{ and } l\neq j}$ (Visit_table(j,l)) +1,
  End For
End If
End While

Algorithm 5. Pseudocode of AHA algorithm.

The second function is Rastrigin function, which is relatively difficult to be optimized because of multiple local optima. Its optimal solution is $x = (0, 0)$ with $f(x) = 0$. The searching behaviors of AHA for this function are shown in Fig. 9. Inspecting this figure, through the special flight skills in the optimization process, AHA can perform a good exploration for the entire search space and converge to the global optimum quickly. After 100 iterations, the best solution found by the proposed algorithm using four hummingbirds is x=(−5.8377E−11, 2.7365E−10) with $f(x) = 0$.

3.4. Conceptual comparison of AHA with ABC

AHA and ABC are a swarm-based, bio-inspired meta-heuristic algorithm, formally, though, they look similar, AHA has its characteristics. ABC models the foraging behaviors of three types of honey bees when seeking the food sources, while AHA focuses more on modeling the memory ability and flight skills of hummingbirds in the foraging process. The optimization process of ABC can be divided into three phases: the employed bee search phase, the onlooker bee search phase, and the scout bee search phase. In the employed bee search phase, each individual in the population updates its position with respect to a randomly chosen individual except itself. In the onlooker bee search phase, each individual in the population updates its position with respect to an individual chosen by a roulette mechanism based on the fitness value. The position updates of individuals in the two phases all adopt a random

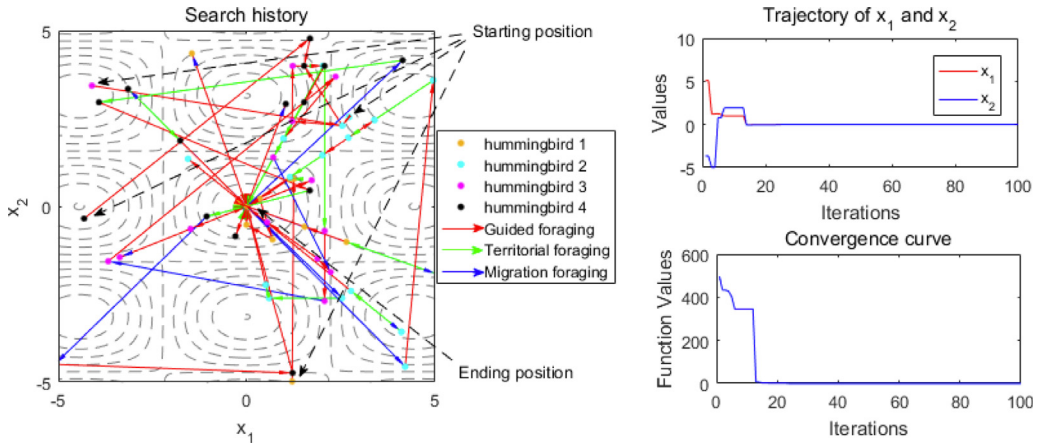

Fig. 9. Searching behaviors of AHA on Rastrigin function.

walk. In the scout bee search phase, the position of the individual achieving a predefined limit value is replaced by a new randomly produced position in the search space. In AHA, there are three phases in the optimization process, including the guided foraging, territorial foraging, and migration foraging phases. In the guided foraging phase, each individual in the population updates its position with respect to an individual chosen by a visit table and its fitness value, a visit table keeps track of how long the positions of other individuals have not been visited by it, thus the position of the individual who has not been visited for the longest time and has the better fitness value becomes its visiting target. In the territorial foraging phase, each individual updates its position by adding a perturbation in its neighboring region. The position updates of individuals in the two phases adopt three fight skills. In the migration foraging phase, when the migration coefficient is satisfied in the iteration process, the position of the individual with the worst fitness value is replaced by a new randomly produced position in the search space. Therefore, based on the above-mentioned analysis, it can be found that ABC and AHA follow totally different approaches when solving optimization problems.

The above description of the algorithm indicates:

- AHA is a bio-inspired meta-heuristic method motivated by the foraging and flights of hummingbirds for solving global optimization problems.
- The algorithm models three foraging strategies of hummingbirds: guided foraging, territorial foraging, and migrating foraging.
- The proposed three flight skills, including omnidirectional, diagonal, and axial flights, are integrated into the strategies of guided foraging, territorial foraging to perform an effective search.
- The visit table records the visit level of each food source for different hummingbirds. AHA needs to maintain the visit table by which each hummingbird may find its own target food source and visit that source.
- The visit table can improve the traverse visit of hummingbirds for all the food sources and reduce the repeated visit to the same food source.
- In the guided foraging strategy, each hummingbird needs to determine its own target food source, which is the food source with the best nectar-refilling rate among the food sources with the same highest visit level.
- The guided foraging strategy is devoted to exploration in the early stage of iterations but focuses on exploitation in the later stage of iterations.
- The territorial foraging strategy contributes to exploitation and migration is dedicated to exploration.
- AHA is very simple to implement and requires that few parameters are adjusted.

In the next section, AHA will be tested on several case studies.

4. Experimental analysis and results

To effectively verify the performance of the AHA algorithm, three different sets of experiments are implemented, and the optimization results provided by AHA are analyzed and compared with those provided by the other optimizers. The first experiment comprises 50 numerical test functions with different kinds of characteristics that are relatively comprehensive to assess the performance of optimizers from multiple perspectives. The second experiment

Table 1
Parameter settings for each algorithm.

Algorithm	Parameter	Value
CS	Mutation probability	0.25
DE	Mutation factor, crossover rate	0.5, 0.5
PSO	Cognitive coefficient, social coefficient, inertia constant	2, 2, decrease from 0.9 to 0
GSA	Gravitational constant, decreasing coefficient	100, 20
ABC	Limit	$n{\cdot}d$
TLBO	Teaching factor	[1, 2]
SHADE	Pbest rate, arc rate	0.1, 2
CMA-ES	Expected initial distance to optimum per coordinate Coordist	5
WOA	Convergence parameter	Linear reduction from 2 to 0
SSA	Leader position update probability	0.5
BOA	Modular modality, power exponent, switch probability	0.01, 0.1, 0.8
AHA	Migration coefficient	$2n$

is the CEC 2014 test suite that is widely employed to test the performance of algorithms in terms of both exploration and exploitation. The third experiment consists of 10 engineering design cases, which are employed to check the effectiveness of the AHA algorithm.

4.1. Experiment 1: 50 benchmark functions

In this experiment, 50 test functions described in Appendix A are employed and their details can be found in [22]. This test set is reasonably large and covers functions with four characteristics: unimodal, multimodal, nonseparable, and separable. The unimodal functions have only one local extremum, while the multimodal functions have more than one. The multimodal characteristic always makes algorithms easy to obtain the local optima. The separable characteristic indicates that the variables of the functions can be split into a product of functions of every variable, but the nonseparable characteristic cannot because of the interrelation among their variables. The nonseparable characteristic tends to lead to the global optimum being difficult to find. These functions are listed in Appendix A, in which 17 functions are unimodal, 33 are multimodal, 36 are nonseparable, and 14 functions are separable.

The performance of AHA is compared with those of eleven different swarm-based algorithms, including CS, DE, PSO, GSA, ABC, TLBO, covariant matrix adaptation with evolution strategy (CMA-ES) [68], success-history based adaptive DE (SHADE) [69], WOA, salp swarm algorithm (SSA) [70], and BOA. The population size and the number of function evaluations (FEs) are set as 50 and 50,000 for all considered optimizers, which are run 30 times for each function. The parameter values of each algorithm are provided in Table 1.

The mean ('Mean') and standard deviations ('Std') of the best-so-far solutions are used to compare all the considered algorithms as two evaluation criteria in this experiment, which are formulated as follows:

$$Mean = \frac{1}{R}\sum_{i=1}^{R} g_i^* \tag{14}$$

$$Std = \sqrt{\frac{1}{R}(g_i^* - Mean)^2} \tag{15}$$

where g_i^* is the best-so-far solution obtained in the ith independent run and R is the number of independent runs. It is obvious that the smaller the values of the two evaluation criteria, more stable and reliable solutions the algorithm can offer. The standard solution (true value) is the globally optimal solution that means there exists no other feasible solution with better objective function values. For a minimum problem, the standard solution is the feasible solution where its function value reaches its minimum value. The smaller the function value of the feasible solution is, the closer the feasible solution is to the standard solution [71].

Table 2
Comparisons of optimization results for 50 test functions (F1–F17).

Fun.	Index	AHA	PSO	TLBO	DE	CS	GSA	ABC	CMA-ES	SHADE	WOA	SSA	BOA
F_1	Mean	−5	−1.66667	−4.966667	−5	−5	−1.7	−5	−0.46667	−4.23333	−5	−3.66667	-2
	Std	0	0.660895	0.182574	0	0	0.466092	0	1.33218	2.9206	0	0.71116	1.94759
F_2	Mean	0	0.066667	0	0.333333	36.3	0	4.966667	0	0	0	5.13333	0
	Std	0	0.253721	0	0.844183	10.299481	0	1.449931	0	0	0	2.93297	0
F_3	Mean	7.32E−292	9.89E−08	7.06E−87	3.38E−14	14.940417	2.23E−17	1.544734	8.13E−10	1.16E−06	2.48E−208	5.22E−09	2.94E−13
	Std	0	4.88E−07	1.02E−86	5.01E−14	3.954443	5.97E−18	0.515346	4.52E−10	4.88E−07	0	8.73E−10	9.63E−14
F_4	Mean	1.67E−288	3.10E−10	1.54E−87	5.82E−15	2.057293	1.71E−16	0.168377	1.43E−10	1.70E−07	3.01E−205	7.48E−04	1.46E−12
	Std	0	6.36E−10	1.87E−87	9.24E−15	0.588976	3.88E−17	4.19E−02	7.36E−11	6.76E−08	0	1.47E−03	4.62E−13
F_5	Mean	6.06E−05	4.12E−02	1.02E−03	2.05E−01	7.77E−02	2.06E−02	0.150642	6.40E−03	1.26E−01	3.67E−04	1.31E−02	7.92E−04
	Std	4.83E−05	1.72E−02	2.96E−04	7.19E−02	1.94E−02	7.12E−03	3.93E−02	1.99E−03	4.07E−02	3.61E−04	5.21E−03	3.32E−04
F_6	Mean	0	0	0	0	1.36E−17	4.94E−28	1.85E−14	0	1.13E−27	1.71E−14	4.00E−16	1.47E−05
	Std	0	0	0	0	3.92E−17	1.90E−27	4.09E−14	0	4.30E−27	4.21E−14	3.48E−16	1.43E−05
F_7	Mean	−1	−1	−1	−1	−1	−0.96667	−1	-1	−0.20236	−1	−1	−0.99999
	Std	0	0	0	0	1.75E−13	0.182574	6.31E−12	0	0.40574	1.02E−10	2.30E−13	4.69E−06
F_8	Mean	7.66E−280	7.59E−94	9.91E−135	7.42E−255	5.50E−25	6.21E−22	1.01E−09	2.46E−315	8.29E−45	0	1.06E−16	4.19E−21
	Std	0	2.76E−93	3.53E−134	0	1.94E−24	6.62E−22	1.27E−09	0	1.19E−44	0	1.49E−16	4.54E−21
F_9	Mean	2.83E−24	7.63E−03	9.99E−06	3.84E−06	1.19E−02	0.697872	4.62E−02	5.81E−24	1.23E−06	0.15092	1.52E−10	9.11E−02
	Std	2.52E−24	6.49E−03	2.26E−05	2.10E−05	2.04E−02	0.933509	5.35E−02	3.18E−24	1.69E−06	0.104465	2.06E−10	9.24E−02
F_{10}	Mean	−50	−50	−50	−50	−50	−50	−50	−50	−50	−50	−50	−49.98911
	Std	6.53E−14	2.96E−14	2.47E−14	2.89E−14	2.67E−10	2.36E−14	1.73E−05	3.95E−14	5.72E−14	7.90E−08	3.94E−12	4.60E−03
F_{11}	Mean	−210	−209.9986	−210	−210	−209.999	−209.909	−206.2352	−210	−209.99999	−209.99989	−210	−205.28953
	Std	2.98E−06	4.71E−03	1.47E−05	3.15E−13	6.21E−04	0.164415	1.851794	4.24E−13	9.38E−06	9.50E−05	1.53E−10	5.6689
F_{12}	Mean	1.78E−262	2.50E−17	1.09E−51	5.28E−33	2.98E−03	4.30E−18	0.137357	4.54887	1.86E−06	3.46E−08	6.71E−12	1.12E−16
	Std	0	6.22E−17	2.35E−51	7.02E−33	1.44E−03	1.76E−18	9.02E−02	11.90241	1.51E−06	1.26E−07	2.57E−12	5.04E−17
F_{13}	Mean	4.25E−294	2.33E−03	8.13E−06	2.11E−04	0.321206	5.85E−03	4.385384	2.16E−03	1.56E−03	2.25E−07	2.38E−02	5.80E−18
	Std	0	1.51E−03	1.74E−05	1.48E−04	0.126297	4.56E−03	2.310504	7.42E−04	8.84E−04	5.28E−07	1.08E−02	2.10E−18
F_{14}	Mean	8.99E−150	7.79E−05	9.40E−44	5.45E−08	19.332082	2.37E−08	3.40E−02	4.58E−05	1.01E−03	3.90E−117	0.16395	3.51E−06
	Std	3.87E−155	1.39E−04	6.87E−44	2.36E−08	4.292123	3.40E−09	6.27E−03	1.77E−05	2.77E−04	1.11E−116	0.33186	3.57E−07
F_{15}	Mean	2.96E−278	1078.0941	2.16E−16	7.526372	3102.901	271.9731	11540.79	0.33573	144.87304	977.52989	5.49E−04	1.64E−13
	Std	0	647.25346	6.13E−16	9.786907	582.7381	75.43658	1588.385	0.39314	29.2366	866.21494	8.73E−04	4.63E−14
F_{16}	Mean	25.065057	49.164576	23.377421	38.588551	784.6473	26.09923	8717.4121	25.95412	26.06047	25.15995	48.64191	28.66781
	Std	0.278139	30.081342	0.703925	23.616923	271.3126	0.201436	2787.6436	33.08219	0.32561	0.2677	48.27414	3.10E−02
F_{17}	Mean	0.666667	1.369245	0.666667	0.692593	9.670616	0.676004	66.23423	0.66667	0.66668	0.66667	0.7314	0.72472
	Std	8.50E−17	1.249854	1.11E−14	0.107241	2.772505	3.00E−02	22.11624	5.87E−06	1.03E−05	9.21E−06	0.1064675	1.36E−02

Table 3
Comparisons of optimization results for 50 test functions (F18–F34).

Fun.	Index	AHA	PSO	TLBO	DE	CS	GSA	ABC	CMA-ES	SHADE	WOA	SSA	BOA
F_{18}	Mean	0.998003	0.998003	0.998003	0.998003	0.998004	3.638354	0.998138	3.00695	0.998003	0.998003	0.998003	0.998003
	Std	0	0	0	0	2.19E−15	2.217956	5.61E−04	2.4873	0	1.36E−13	1.82E−16	5.42E−08
F_{19}	Mean	0.397887	0.397887	0.397887	0.397887	0.397887	0.397887	0.397887	0.397887	0.43609	0.397887	0.397887	0.39791
	Std	0	0	0	0	2.65E−14	0	7.36E−09	0	9.18E−02	1.72E−09	2.88E−15	2.63E−05
F_{20}	Mean	0	0	0	0	0	0	5.10E−10	0	61.90491	0	3.52E−12	0
	Std	0	0	0	0	0	0	5.13E−10	0	102.66111	0	4.70E−12	0
F_{21}	Mean	0	0	0	0	1.38E−23	1.79E−20	2.36E−10	0	0	4.19E−07	2.52E−15	5.59E−06
	Std	0	0	0	0	2.71E−23	2.19E−20	3.25E−10	0	0	3.25E−07	3.27E−15	6.65E−06
F_{22}	Mean	0	30.845624	12.679728	153.2381	109.4123	14.6259	188.6345	1.65E+02	1.15E+02	0	33.79541	5.79935
	Std	0	7.6342136	5.5850898	32.147688	13.43076	3.265065	12.28813	8.99333	9.03763	0	14.36489	31.76315
F_{23}	Mean	−12409.83	−7414.212	−6967.84	−5410.479	−8240.78	−2799.41	−4669.02	−4.37E+03	−6.87E+03	−12373.69	−7.75E+03	−4.64E+03
	Std	225.97788	703.19062	883.35773	626.77109	183.6841	401.8812	321.0048	206.39451	354.24722	477.69677	622.8568	346.07855
F_{24}	Mean	−1.801303	−1.801303	−1.801303	−1.801303	−1.801303	−1.801303	−1.801303	−1.80127	−1.801303	−1.801303	−1.801303	−1.801206
	Std	9.03E−16	9.03E−16	9.03E−16	9.03E−16	9.03E−16	9.53E−16	2.82E−15	1.79E−04	9.03E−16	1.57E−10	7.49E−15	1.48E−04
F_{25}	Mean	−4.687658	−4.676521	−4.630384	−4.672345	−4.68747	−4.58342	−4.68325	−4.65679	−4.687658	−4.20252	−4.47196	−4.22945
	Std	1.61E−15	1.88E−02	6.22E−02	2.05E−02	1.73E−04	7.45E−02	8.45E−03	6.42E−03	1.66E−14	0.47952	0.26152	0.13664
F_{26}	Mean	−9.659194	−9.436739	−9.3722	−9.350663	−8.32567	−9.29025	−8.39637	−6.408	−9.16678	−7.37496	−7.90679	−5.84003
	Std	1.88E−02	0.164514	0.174254	0.377745	0.261795	0.196015	0.319772	0.757190	0.100900	0.93505	0.78449	0.27924
F_{27}	Mean	0	0	0	0	6.40E−12	1.12E−02	3.99E−06	0	5.97E−03	0	1.41E−16	0
	Std	0	0	0	0	2.56E−11	1.24E−02	5.31E−06	0	3.24E−02	0	2.57E−16	0
F_{28}	Mean	−1.031628	−1.031628	−1.031628	−1.031628	−1.031628	−1.031628	−1.031628	−1.03163	−1.03163	−1.03163	−1.03163	−1.03163
	Std	6.65E−16	6.78E−16	6.78E−16	6.78E−16	5.13E−16	5.68E−16	7.78E−11	6.78E−16	6.71E−16	1.77E−14	1.56E−15	1.87E−06
F_{29}	Mean	0	0	0	0	0	0	2.92E−09	3.91E−02	79.93775	0	2.63E−12	1.85E−17
	Std	0	0	0	0	0	0	3.50E−09	0.21426	110.07691	0	2.95E−12	2.66E−17
F_{30}	Mean	0	0	0	0	0	0	3.46E−07	0	0	0	1.18E−12	0
	Std	0	0	0	0	0	0	3.63E−07	0	0	0	1.38E−12	0
F_{31}	Mean	−186.7309	−186.7309	−186.7309	−186.7309	−186.7309	−185.1421	−186.7161	−186.56881	−186.7309	−186.7309	−186.7309	−186.69803
	Std	2.30E−14	4.26E−14	2.04E−14	3.12E−14	1.44E−07	1.482282	1.83E−02	0.6186	3.98E−05	1.96E−07	4.28E−12	2.95E−02
F_{32}	Mean	3	3	3	3	3	3	3	3.00097	3	3	3	3.00009
	Std	1.33E−15	2.24E−15	1.26E−15	1.91E−15	1.44E−15	2.37E−15	1.47E−10	5.30E−03	1.66E−15	1.88E−08	3.87E−14	8.05E−05
F_{33}	Mean	3.07E−04	3.23E−04	3.08E−04	3.08E−04	4.01E−04	1.89E−03	5.61E−04	5.35E−04	3.08E−04	4.96E−04	4.82E−04	3.50E−04
	Std	1.71E−19	3.33E−05	1.20E−14	1.13E−19	7.66E−05	5.82E−04	6.53E−05	3.21E−04	2.58E−12	2.81E−04	2.12E−04	2.81E−05
F_{34}	Mean	−10.1532	−8.387923	−10.1532	−9.816371	−10.1532	−6.51465	−10.04147	−7.19831	−10.1532	−10.1532	−9.98479	−10.07206
	Std	7.17E−15	2.5745855	6.40E−15	1.281842	1.56E−07	3.584548	0.612238	3.50661	6.56E−15	2.50E−06	0.92244	5.29E−02

The evaluation results of the best-so-far solutions are provided on each function in Tables 2–4, in which AHA offers the best results of all the algorithms on 39 functions and is competitive as well on the other functions. Specifically, although AHA is similar to DE, the performance of AHA, DE and its variant SHADE is quite different. On 41 out of 50 benchmark functions the mean solutions provided by AHA are the best. On 25 out of 50 benchmark functions the mean solutions provided by DE are the best. On 17 out of 50 benchmark functions the mean solutions provided by SHADE are the best. Moreover, AHA, DE and SHADE provide the same best mean solutions on some functions such as F10, F18, F21, F24, F30, F31 and F32, whereas AHA performs significantly better than DE and SHADE on many functions such as F3, F4, F5, F9, F12, F13, F14, F15, F17, F22, F23, F26, F33, F41, F42 and F43. To effectively evaluate the overall performance of AHA, the Wilcoxon signed-rank test [72] is employed for

Table 4
Comparisons of optimization results for 50 test functions (F35–F50).

Fun.	Index	AHA	PSO	TLBO	DE	CS	GSA	ABC	CMA-ES	SHADE	WOA	SSA	BOA
F35	Mean	−10.40294	−9.87415	−10.22344	−10.40294	−10.40294	−10.40294	−10.40294	−7.41732	−10.40294	−10.40294	−9.82871	−10.29099
	Std	1.44E−15	1.613482	0.983192	1.68E−15	1.39E−06	6.60E−16	1.27E−13	1.92462	9.90E−16	1.23E−05	1.76503	7.29E−02
F36	Mean	−10.53641	−10.00033	−10.53641	−10.53641	−10.53641	−10.53641	−10.53641	−8.16297	−10.53641	−10.53617	−10.35772	−10.38754
	Std	1.78E−15	1.635722	2.06E−15	1.78E−15	3.86E−06	1.75E−15	4.01E−13	3.69341	2.29E−15	1.30E−03	0.97874	0.10201
F37	Mean	4.94E−02	0.605883	2.18E−02	9.45E−03	8.47E−03	7.574478	0.379297	0.162175	0.13143	0.8788	3.57E−03	1.25213
	Std	0.120376	0.739731	4.04E−02	2.29E−02	5.77E−03	5.159717	0.316325	3.92E−02	9.70E−02	1.23697	2.96E−03	0.93553
F38	Mean	1.45E−04	1.19E−03	1.88E−03	1.41E−04	7.55E−04	4.59E−02	1.60E−02	6.68E−04	9.70E−03	4.36E−01	1.33E−02	2.40E−02
	Std	1.79E−04	2.91E−03	2.36E−03	1.35E−04	5.75E−04	4.65E−02	1.04E−02	9.74E−04	6.24E−03	4.07E−01	2.48E−02	1.45E−02
F39	Mean	−3.862782	−3.862782	−3.862782	−3.862782	−3.862782	−3.862782	−3.862782	−3.862782	−3.862782	−3.86277	−3.86278	−3.86254
	Std	2.71E−15	2.71E−15	2.71E−15	2.71E−15	2.32E−15	2.43E−15	2.03E−15	2.71E−15	2.71E−15	1.89E−05	4.82E−15	1.23E−04
F40	Mean	−3.310106	−3.254622	−3.318032	−3.302181	−3.321997	−3.322	−3.322	−3.30989	−3.322	−3.2541	−3.27337	−3.30731
	Std	3.63E−02	5.99E−02	2.17E−02	4.51E−02	4.69E−07	1.36E−15	4.80E−15	3.69E−02	1.36E−15	6.04E−02	6.06E−02	1.19E−02
F41	Mean	0	1.67E−02	0	3.70E−03	1.150341	4.289197	0.934173	4.65E−09	1.07E−05	1.74E−03	9.11E−03	0
	Std	0	1.92E−02	0	9.72E−03	4.11E−02	2.066729	5.82E−02	2.09E−09	2.12E−05	4.60E−03	8.74E−03	0
F42	Mean	8.88E−16	9.97E−02	6.57E−15	3.99E−08	9.189665	3.38E−09	0.655145	8.86E−06	3.55E−04	2.90E−15	1.2144	3.63E−07
	Std	0	0.380955	1.77E−15	1.85E−08	1.819592	4.02E−10	0.216113	2.60E−06	7.07E−05	2.22E−15	0.97792	7.41E−08
F43	Mean	4.15E−07	0.1774656	3.46E−03	1.04E−02	3.121021	4.32E−02	37.79774	4.19E−11	4.35E−07	4.35E−06	1.37761	7.44E−02
	Std	5.63E−07	0.2800774	1.89E−02	3.16E−02	0.691772	6.97E−02	20.68029	2.50E−11	3.03E−08	1.61E−06	1.37411	2.72E−02
F44	Mean	0.669596	7.69E−03	4.26E−02	1.47E−03	7.714342	2.17E−18	5657.875	5.98E−10	4.91E−07	1.94E−03	2.93E−03	4.17E−01
	Std	0.547843	1.45E−02	4.90E−02	3.80E−03	2.177475	5.70E−19	5582.683	4.54E−10	2.30E−07	8.19E−03	4.94E−03	1.40E−01
F45	Mean	−1.080938	−1.080938	−1.080938	−1.080938	−1.080938	−1.060773	−1.080938	−1.08066	−1.080938	−1.080938	−1.080938	−1.08007
	Std	4.52E−16	2.46E−11	4.52E−16	4.52E−16	6.45E−16	2.35E−02	1.38E−06	9.58E−04	4.52E−16	9.74E−16	1.96E−15	1.48E−03
F46	Mean	−1.331229	−1.499999	−1.482166	−1.499999	−1.476211	−0.695269	−1.5	−1.26141	−1.5	−0.86853	−1.19567	−0.90248
	Std	0.2871761	6.78E−16	9.77E−02	6.78E−16	9.74E−02	0.134028	2.71E−10	0.29157	3.14E−10	0.25593	0.35396	0.19274
F47	Mean	−0.568135	−0.662263	−0.605597	−1.128604	−0.75937	−0.10345	−1.09223	−0.73411	−1.16257	−0.35923	−0.49452	−0.19268
	Std	0.203332	0.181552	0.270130	0.405098	9.88E−02	0.111923	0.233308	0.3812	0.28561	0.15903	0.26391	9.87E−02
F48	Mean	0	0	0	0	2.30E−13	2.51E−17	3.80E−10	0.21623	17.16006	1.60E−11	8.10E−13	4.96E−03
	Std	0	0	0	0	4.49E−13	2.15E−17	7.57E−10	0.69008	25.83799	5.57E−11	8.72E−13	5.06E−03
F49	Mean	1.99E−14	24.996503	6.30E−04	7.40E−29	0.172079	627.2041	0.206423	6.38E−04	1.32E−03	119.49457	3.30E−04	112.36312
	Std	8.38E−14	38.797106	3.38E−03	9.90E−29	0.20066	1066.61	0.275318	2.38E−03	1.62E−03	432.23037	8.73E−04	54.99002
F50	Mean	26.091881	1532.5689	191.36318	3.02667	509.7585	8576.55	21.00176	1.15E+04	20.08336	5.27E+03	13.46566	5.55E+03
	Std	40.154103	2755.4617	446.40356	14.34583	170.7582	9659.197	5.291822	1.82E+04	76.03819	6.89E+03	46.29551	1.57E+03

Table 5
Statistical results of Wilcoxon signed-rank test for AHA versus other algorithms.

Function characteristics	AHA vs PSO (+/=/−)	AHA vs TLBO (+/=/−)	AHA vs DE (+/=/−)	AHA vs CS (+/=/−)	AHA vs GSA (+/=/−)	AHA vs ABC (+/=/−)
US	4/1/0	3/2/0	4/1/0	4/1/0	4/1/0	4/1/0
UN	9/3/0	8/3/1	8/2/2	12/0/0	10/2/0	12/0/0
MS	4/5/0	4/5/0	4/5/0	7/2/0	6/3/0	9/0/0
MN	11/12/1	5/17/2	5/15/4	16/7/1	15/8/1	21/2/1
Total	28/21/1	20/27/3	21/23/6	39/10/1	35/14/1	46/3/1

Function characteristics	AHA vs CMA-ES (+/=/−)	AHA vs SHADE (+/=/−)	AHA vs WOA (+/=/−)	AHA vs SSA (+/=/−)	AHA vs BOA (+/=/−)
US	4/1/0	3/2/0	3/2/0	5/0/0	4/1/0
UN	5/2/5	12/0/0	10/1/1	11/0/1	12/0/0
MS	5/3/1	6/3/0	6/3/0	9/0/0	8/1/0
MN	7/12/5	13/8/3	19/4/1	21/1/2	20/4/0
Total	21/18/11	34/13/3	38/10/2	46/1/3	44/6/0

better comparison. With the results of 50 functions from 30 runs of each algorithm, the Wilcoxon signed-rank test can check whether AHA is better than the other competitors with a significance level of $a = 0.05$. The statistical results of the significance difference using the Wilcoxon signed-rank test are summarized in Table 5. In Table 5, '=' denotes that there is no statistically significant difference between AHA and the comparative method, '+' means that the null hypothesis is rejected and AHA statistically outperforms another algorithm, and '−' is for the reverse.

As can be seen in Table 5, AHA displays superior performance to the other algorithms on the US (unimodal and separable) functions. AHA performs considerably better than the others on the UN (unimodal and nonseparable) functions. The results on the US and UN functions demonstrate the superior exploration ability of AHA. It is also seen that AHA performs better than the others on MS (multimodal and separable) functions; the results of all UN and MS functions provided by ABC algorithm are inferior to those provided by AHA. For MN (multimodal and nonseparable) functions, there is no significant difference between AHA and TLBO on 17 functions. TLBO performs better than AHA on two MN functions, while AHA is better than TLBO on five MN functions. There is no significant difference between AHA and DE on 15 functions. AHA is better than DE on five MN functions and is worse than DE on four MN functions. In addition, there is no significant difference between AHA and PSO on

Table 6
Friedman test of all compared algorithms for 50 functions.

	AHA	PSO	TLBO	DE	CS	GSA	ABC	CMA-ES	SHADE	WOA	SSA	BOA
Sum of ranks	155	321.5	206	224.5	365.5	373.5	457	343	301.5	342	392	418.5
Mean of ranks	3.1	6.43	4.12	4.49	7.31	7.47	9.14	6.86	6.03	6.84	7.84	8.37
Overall ranks	1	5	2	3	8	9	12	7	4	6	10	11

12 MN functions; AHA is superior to PSO on 11 MN functions and is inferior to PSO on only one MN function. The results on MN functions suggest that AHA has a merit with respect to exploration and obviously surpasses all the other algorithms.

To rank the performance of the considered algorithms, the Friedman test [73,74] is employed in this experiment. This test not only answers whether there is a significant difference between AHA and the comparative optimizers but also ranks the value of each algorithm from the lowest to the highest. The better an algorithm is, the lower its rank is. This test is carried out on the average solutions provided by AHA and other meta-heuristics, and the ranks resulted from this test are obtained in Table 6. Inspecting this table, obviously, the AHA algorithm is highlighted as the best method from the comparisons, with the mean ranks of 3.1 for Friedman test.

Most optimization algorithms have good optimization ability for low-dimensional problems. However, for high-dimensional problems, some algorithms become unsatisfactory since the size of the search space increases exponentially with dimensionality. To verify the scalability of AHA to various high-dimensional functions, 14 variable-dimensional functions out of 50, including functions F2 (Step), F3 (Sphere), F4 (SumSquares), F5 (Quartic), F14 (Schwefel 2.22), F15 (Schwefel 1.2), F16 (Rosenbrock), F17 (Dixon–Price), F22 (Rastrigin), F23 (Schwefel), F41 (Griewank), F42 (Ackley), F43 (Penalized), and F44 (Penalized2), are employed. The number of dimensions of these employed functions is increased from 30 to 300 with a step of 15. The mean best-so-far solutions provided by all the methods for each variable-dimensional function over 30 runs and 50,000 FEs are compared for each dimension.

The scalability comparisons of AHA versus the other algorithms in tackling these employed functions are provided in Fig. 10. Based on the figure, the results provided by AHA demonstrate much slower degrades than those provided by the other algorithms and remain consistently superior on most functions as the number of dimensions grows. Even when the function dimensions are very high, the solution quality of AHA is considerably competitive. Although AHA performs slightly worse than TLBO with respect to lower dimensions on functions F43 and F44, with an increase of dimensions, AHA can achieve almost the same searching quality as TLBO, which is better than that of all the other optimizers. In addition, the convincing results suggest that the AHA algorithm is also able to strike a fine balance between the exploratory and exploitative search, even on multimodal functions with multiple dimensions.

4.2. Experiment 2: CEC 2014 test suite

In this experiment, a popular benchmark suite, i.e., CEC 2014 test suite [75], is employed to check whether AHA is capable to provide more superior optimization performance than the peer algorithms. Because of space constraints, only one dimension case, i.e., D=30, is considered. The CEC 2014 test suite described in Appendix B consists of 3 unimodal functions (CF1–CF3), 13 multimodal functions (CF4–CF16), 6 hybrid functions (CF17–CF22), and 8 composition functions (CF23–CF30). To maintain consistency, the 11 peer algorithms used in the first experiment are also employed to this experiment. The maximum number of function evaluations and the population size for all the considered algorithms are set to 25,000 and 50, respectively. Other parameter settings of these algorithms are summarized in Table 1. Every algorithm runs 30 times and the results are based on these 30 runs in average. The results obtained for AHA versus other optimizers from the CEC 2014 test suite are listed in Tables 7 and 8. From the two tables, AHA can offer the best results compared to other optimizers on 11 functions, i.e., CF8, CF10, CF11, CF16, CF23–CF25, and CF27–CF30, in tackling 36.7% of the test suite, followed by SHADE who offers the best results on 8 functions, i.e., CF2–CF4, CF7, CF17, and CF19–CF21, in tackling 26.7% of the test suite. Furthermore, TLBO also displays the best performance on 6 functions, i.e., CF1, CF6, CF9, CF13, CF18, and CF22, in tackling 20.0% of the test suite. However, PSO, DE, CMA-ES, WOA and BOA do not show the best performance on any function.

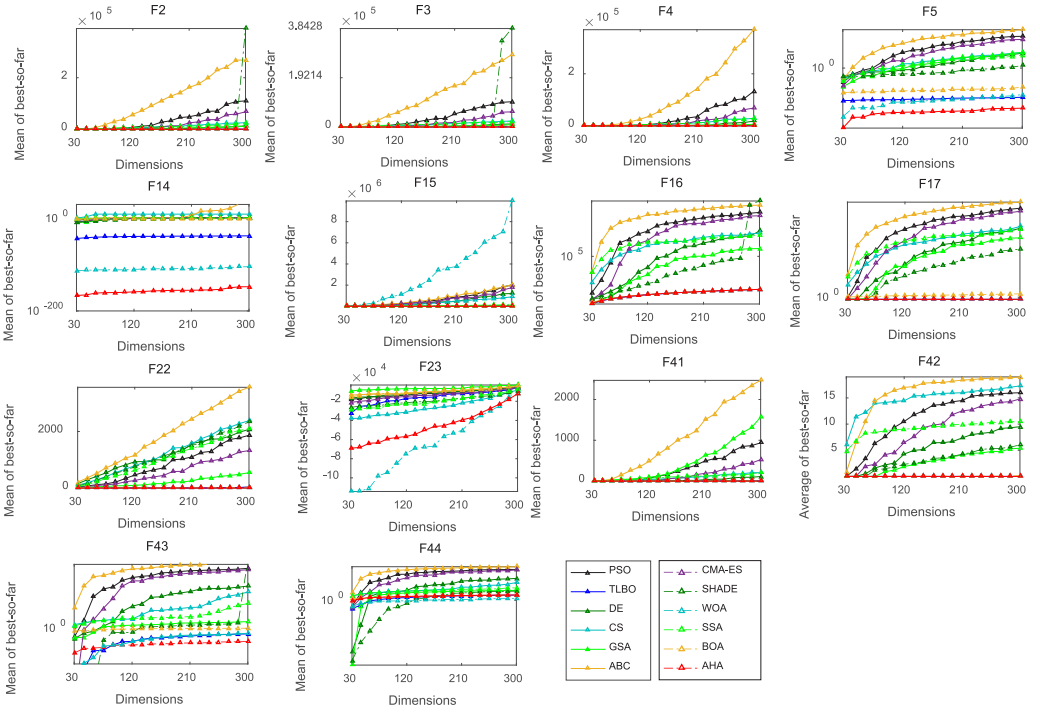

Fig. 10. Scalability comparisons of algorithms for unimodal and multimodal functions.

Table 7
Results on unimodal and multimodal functions in CEC 2014.

Fun.	Index	AHA	PSO	TLBO	DE	CS	GSA	ABC	CMA-ES	SHADE	WOA	SSA	BOA
CF_1	Mean	2.27E+07	2.14E+08	6.81E+06	1.10E+08	9.62E+07	4.20E+08	2.22E+08	1.89E+08	9.91E+06	2.00E+08	2.58E+07	3.24E+08
	Std	1.22E+07	8.66E+07	3.43E+06	2.61E+07	2.40E+07	8.23E+07	4.81E+07	8.33E+07	2.82E+06	9.74E+07	1.33E+07	1.14E+08
CF_2	Mean	9.42E+06	5.54E+09	9.85E+05	2.07E+05	3.46E+09	2.33E+10	3.07E+08	3.89E+09	1.01E+04	3.32E+09	1.03E+04	3.30E+10
	Std	6.98E+06	4.17E+09	1.16E+06	6.28E+04	8.18E+08	3.26E+09	7.11E+07	2.27E+09	4.40E+03	1.65E+09	8.60E+03	7.10E+09
CF_3	Mean	6.47E+03	2.55E+04	4.27E+04	1.91E+03	2.20E+04	8.73E+04	8.00E+04	2.66E+04	5.08E+02	9.81E+04	6.78E+04	6.80E+04
	Std	3.59E+03	1.78E+04	6.45E+03	1.35E+03	4.00E+03	6.51E+03	1.36E+04	1.85E+04	1.26E+02	4.52E+04	1.65E+04	9.06E+03
CF_4	Mean	5.67E+02	2.00E+03	5.63E+02	5.91E+02	8.05E+02	2.83E+03	7.95E+02	1.94E+03	5.01E+02	1.01E+03	5.60E+02	4.08E+03
	Std	4.77E+01	5.46E+02	3.69E+01	1.98E+01	6.68E+01	3.37E+02	4.52E+01	3.64E+02	2.92E+01	1.90E+02	4.12E+01	1.04E+03
CF_5	Mean	520.1335	521.0105	521.0691	520.8434	521.0028	519.9992	521.0542	520.9916	521.0091	520.7941	519.9999	521.0673
	Std	9.33E−02	8.82E−02	4.66E−02	5.90E−02	6.84E−02	4.90E−04	5.95E−02	8.92E−02	4.54E−02	1.15E−01	1.11E−04	5.97E−02
CF_6	Mean	6.17E+02	6.29E+02	6.15E+02	6.30E+02	6.33E+02	6.32E+02	6.32E+02	6.28E+02	6.19E+02	6.38E+02	6.20E+02	6.35E+02
	Std	2.68E+00	3.82E+00	2.46E+00	1.19E+00	1.07E+00	2.74E+00	1.37E+00	4.93E+00	4.90E+00	2.78E+00	3.33E+00	1.36E+00
CF_7	Mean	701.0583	787.3136	700.7527	700.2518	718.6952	956.9402	703.4847	790.1209	700.0042	716.6369	700.0133	1007.0856
	Std	7.40E−02	3.23E+01	2.61E−01	5.45E−02	5.08E+00	3.35E+01	4.94E−01	3.47E+01	8.88E−03	4.60E+00	9.59E−03	6.95E+01
CF_8	Mean	8.48E+02	9.10E+02	8.67E+02	8.81E+02	9.87E+02	9.45E+02	1.02E+03	9.17E+02	9.10E+02	1.03E+03	9.42E+02	1.11E+03
	Std	1.75E+01	3.06E+01	1.39E+01	7.35E+00	1.62E+01	1.01E+01	1.15E+01	3.69E+01	1.01E+01	4.63E+01	3.04E+01	1.00E+01
CF_9	Mean	1.05E+03	1.11E+03	1.01E+03	1.09E+03	1.14E+03	1.07E+03	1.15E+03	1.10E+03	1.06E+03	1.19E+03	1.05E+03	1.22E+03
	Std	3.09E+01	4.59E+01	3.14E+01	8.33E+00	2.44E+01	1.44E+01	1.09E+01	4.51E+01	8.42E+00	4.42E+01	4.04E+01	1.64E+01
CF_{10}	Mean	1.67E+03	3.03E+03	5.46E+03	3.56E+03	4.92E+03	4.91E+03	7.48E+03	3.56E+03	5.04E+03	5.76E+03	4.47E+03	8.30E+03
	Std	3.96E+02	7.69E+02	1.32E+03	2.79E+02	1.98E+02	3.60E+02	2.55E+02	9.45E+02	5.22E+02	6.58E+02	6.58E+02	3.97E+02
CF_{11}	Mean	3.89E+03	7.90E+03	8.42E+03	7.44E+03	5.99E+03	5.61E+03	8.63E+03	7.94E+03	7.43E+03	7.05E+03	4.88E+03	8.74E+03
	Std	5.14E+02	7.72E+02	2.94E+02	2.34E+02	2.04E+02	5.07E+02	2.81E+02	8.24E+02	3.71E+02	7.58E+02	6.86E+02	3.11E+02
CF_{12}	Mean	1200.2553	1202.5970	1203.1427	1201.8762	1201.6467	1200.0275	1203.0937	1202.7605	1202.4971	1202.3678	1200.6903	1203.1887
	Std	1.45E−01	5.49E−01	3.75E−01	1.94E−01	2.00E−01	1.31E−02	4.56E−01	5.85E−01	3.14E−01	6.49E−01	3.74E−01	5.06E−01
CF_{13}	Mean	1300.4547	1302.7626	1300.4339	1300.5226	1300.4463	1304.4934	1300.6252	1302.8761	1300.5450	1300.6350	1300.4914	1304.9885
	Std	1.20E−01	5.83E−01	8.87E−02	6.11E−02	5.73E−02	3.90E−01	8.36E−02	4.65E−01	4.32E−02	3.54E−01	1.08E−01	6.66E−01
CF_{14}	Mean	1400.3336	1447.0221	1400.2649	1400.3382	1401.0363	1504.4364	1400.2499	1445.3904	1400.2850	1405.0044	1400.3555	1521.5841
	Std	1.51E−01	1.63E+01	4.84E−02	3.15E−02	8.93E−01	1.39E+01	6.97E−02	1.48E+01	3.86E−02	5.95E+00	1.53E−01	2.57E+01
CF_{15}	Mean	1517.8706	1565.4433	1527.2681	1519.0095	1835.4771	6393.9443	1600.5196	1602.8043	1514.3350	1972.8089	1513.4816	16155.6062
	Std	9.43E+00	6.82E+01	6.09E+00	9.51E−01	2.59E+02	2.61E+03	3.99E+01	2.98E+02	9.37E−01	4.83E+02	3.89E+00	1.60E+04
CF_{16}	Mean	1610.9303	1612.9340	1612.7586	1612.6619	1613.1027	1613.7006	1613.0812	1612.9198	1612.8083	1613.1263	1612.4193	1613.0991
	Std	5.95E−01	2.63E−01	3.01E−01	2.18E−01	1.80E−01	3.01E−01	2.12E−01	2.93E−01	2.66E−01	3.74E−01	4.07E−01	1.84E−01

To analyze the significant differences between the results of AHA and other competitors, the Wilcoxon signed-rank test with 5% significance level is implemented here [72]. The results of the Wilcoxon signed-rank test with 5% significance level of each algorithm for every function are described in Tables 9–11 in detail. The statistical results of the Wilcoxon signed-rank test are shown in Table 12. As shown in Table 12, the performance of AHA on unimodal functions is inferior to that of TLBO, DE and SHADE, and, the performance of TLBO and SHADE is better than that of AHA on hybrid functions, followed by other algorithms. However, AHA achieves a remarkably

Table 8
Results on hybrid and composition functions in CEC 2014.

Fun.	Index	AHA	PSO	TLBO	DE	CS	GSA	ABC	CMA-ES	SHADE	WOA	SSA	BOA
CF_{17}	Mean	1.32E+06	7.64E+06	4.29E+05	5.70E+06	1.80E+06	3.44E+07	5.36E+06	6.70E+06	2.88E+05	1.94E+07	1.57E+06	5.39E+06
	Std	9.18E+05	5.44E+06	2.67E+05	2.55E+06	6.49E+05	8.06E+06	1.42E+06	4.99E+06	1.72E+05	1.39E+07	1.41E+06	3.80E+06
CF_{18}	Mean	3.32E+03	1.87E+06	2.79E+03	3.97E+05	4.74E+07	1.98E+05	1.30E+04	1.51E+06	1.00E+04	1.28E+06	6.67E+03	1.12E+08
	Std	1.70E+03	5.71E+06	1.18E+03	3.31E+05	2.70E+07	1.07E+06	1.54E+04	5.49E+06	6.24E+03	1.49E+06	4.70E+03	1.43E+08
CF_{19}	Mean	1.93E+03	1.97E+03	1.92E+03	1.92E+03	1.93E+03	2.10E+03	1.92E+03	1.97E+03	1.91E+03	1.99E+03	1.92E+03	2.05E+03
	Std	2.83E+01	4.53E+01	2.41E+01	3.31E+00	8.47E+00	2.36E+01	1.31E+00	3.03E+01	7.83E−01	2.66E+01	1.59E+01	4.57E+01
CF_{20}	Mean	1.85E+04	3.20E+04	2.00E+04	9.54E+03	1.48E+04	2.80E+05	4.25E+04	3.61E+04	3.47E+03	1.14E+05	2.86E+04	9.98E+04
	Std	7.19E+03	1.93E+04	5.94E+03	3.02E+03	5.56E+03	1.28E+05	1.51E+04	2.41E+04	2.02E+03	7.68E+04	1.39E+04	5.02E+04
CF_{21}	Mean	2.91E+05	6.57E+05	1.71E+05	1.02E+06	1.73E+05	1.63E+07	1.06E+06	1.06E+06	4.95E+04	6.48E+06	3.05E+05	2.22E+06
	Std	2.17E+05	4.94E+05	1.09E+05	5.32E+05	5.86E+04	4.06E+06	5.06E+05	1.36E+06	4.13E+04	3.91E+06	2.27E+05	1.07E+06
CF_{22}	Mean	2.85E+03	3.03E+03	2.60E+03	2.70E+03	2.71E+03	3.70E+03	2.89E+03	3.04E+03	2.65E+03	3.12E+03	2.67E+03	3.63E+03
	Std	1.94E+02	2.09E+02	1.40E+02	1.36E+02	8.68E+01	5.65E+02	1.41E+02	2.64E+02	8.48E+01	2.95E+02	2.20E+02	2.02E+02
CF_{23}	Mean	2500.0000	2657.4232	2615.3530	2615.6592	2632.8243	2573.1800	2621.9003	2663.2842	2615.2475	2694.1986	2633.2248	2500.0001
	Std	0.00E+00	1.73E+01	1.48E−01	1.30E−01	3.88E+00	1.15E+02	1.33E+00	1.65E+01	2.65E−03	1.90E+01	8.08E+00	2.01E−04
CF_{24}	Mean	2600.0000	2639.1036	2600.0800	2628.1350	2661.0747	2620.0005	2648.2497	2636.3027	2624.9547	2609.0670	2639.0200	2600.0001
	Std	0.00E+00	8.11E+00	1.32E−02	6.18E−01	2.78E+00	5.73E+00	3.37E+00	6.10E+00	9.92E−01	4.80E+00	8.32E+00	1.87E−05
CF_{25}	Mean	2700.0000	2727.5068	2700.0159	2727.0606	2720.8028	2705.6276	2732.6953	2723.8740	2711.2547	2721.1338	2715.0444	2700.0001
	Std	0.00E+00	9.40E+00	8.70E−02	3.31E+00	2.10E+00	2.69E+00	3.65E+00	6.90E+00	1.29E+00	2.40E+01	3.05E+00	3.03E−06
CF_{26}	Mean	2763.5233	2713.5203	2713.7514	2700.6477	2700.4765	2793.1042	2707.7046	2703.2826	2700.8536	2736.6828	2700.5293	2740.4396
	Std	4.88E+01	3.09E+01	3.44E+01	1.17E−01	7.02E−02	2.16E+01	1.96E+01	1.08E+00	6.37E−02	7.95E+01	1.46E−01	4.61E+01
CF_{27}	Mean	2900.0000	3546.1570	3271.3167	3366.2782	3256.9408	4557.5228	3502.8503	3470.9875	3108.8516	3893.3813	3194.4532	3273.4932
	Std	0.00E+00	2.37E+02	1.25E+02	8.35E+01	3.82E+01	3.96E+02	1.22E+02	2.29E+02	8.07E+01	3.00E+02	1.64E+02	4.82E+01
CF_{28}	Mean	3000.0000	7302.5264	3938.6520	4007.9786	3990.5678	5215.9261	4357.7299	7445.9885	3929.0812	5687.5834	4653.9440	6575.4332
	Std	0.00E+00	6.43E+02	1.42E+02	7.00E+01	7.45E+01	8.98E+02	1.18E+02	5.62E+02	7.64E+01	6.58E+02	5.52E+02	6.00E+02
CF_{29}	Mean	3.10E+03	2.34E+07	4.69E+03	3.14E+04	6.27E+05	6.25E+06	3.78E+05	2.85E+07	1.26E+04	1.16E+07	1.35E+04	1.55E+06
	Std	0.00E+00	2.69E+07	9.96E+02	7.35E+03	4.08E+05	2.02E+07	1.97E+05	3.16E+07	2.41E+03	7.58E+06	5.47E+03	3.95E+06
CF_{30}	Mean	4.45E+03	1.90E+05	6.98E+03	2.21E+04	3.78E+04	2.57E+06	4.70E+04	1.01E+05	8.44E+03	3.23E+05	6.39E+04	4.92E+05
	Std	3.53E+03	1.87E+05	1.52E+03	4.48E+03	1.05E+04	6.84E+05	1.32E+04	6.05E+04	9.29E+02	2.02E+05	6.13E+04	2.02E+05

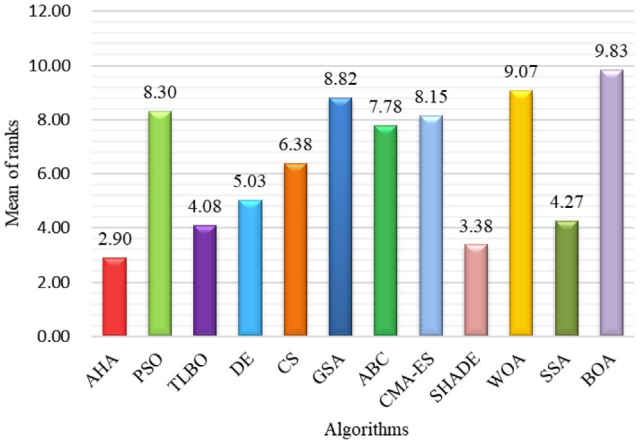

Fig. 11. Mean ranks of the algorithms.

superior performance in dealing with multimodal and composition functions accounting for 70% of the test suite compared to all other optimizers. The results of the Wilcoxon signed-rank test demonstrate that AHA offers the superior performance in terms of solution quality when tackling the CEC test suite.

To further illustrate the overall performance among all 12 competitors, the Friedman test is carried out in this part [73]. The Friedman test can rank the algorithms based on their performance for each problem separately. The performance rank of each algorithm for each function is achieved in Table 13. Inspecting Table 13, among 30 benchmark functions, AHA ranks first for 11 functions and ranks second for 3 functions. Fig. 11 illustrates that the mean ranks of all the algorithms for the CEC test suite. From Fig. 11, the mean rank of AHA is 2.90 which indicates AHA has the best overall performance in dealing with these complicated problems. The results of Friedman test again prove the superiority of AHA over the other considered optimizers.

4.3. Experiment 3: 10 engineering cases

In this subsection, AHA is evaluated with ten constrained engineering design cases, which provide diverse combination of difficulties: constraints, mixed integer, etc. The characteristics of these engineering cases are

Table 9
Wilcoxon signed rank test results for PSO, TLBO, DE and CS versus AHA.

Fun.	PSO vs AHA				TLBO vs AHA				DE vs AHA				CS vs AHA			
	p-value	T+	T−	Winner	*p*-value	T+	T−	Winner	*p*-value	T+	T−	Winner	*p*-value	T+	T−	Winner
CF_1	1.73E−06	0	465	+	2.88E−06	460	5	−	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_2	1.73E−06	0	465	+	1.92E−06	464	1	−	1.73E−06	465	0	−	1.73E−06	0	465	+
CF_3	1.92E−06	1	464	+	1.73E−06	0	465	+	3.88E−06	457	8	−	1.73E−06	0	465	+
CF_4	1.73E−06	0	465	+	8.29E−01	243	222	=	4.39E−03	94	371	+	1.73E−06	0	465	+
CF_5	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_6	1.73E−06	0	465	+	1.31E−03	353	112	−	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_7	1.73E−06	0	465	+	1.80E−05	441	24	−	1.73E−06	465	0	−	1.73E−06	0	465	+
CF_8	1.92E−06	1	464	+	4.20E−04	61	404	+	3.18E−06	6	459	+	1.73E−06	0	465	+
CF_9	3.41E−05	31	434	+	1.36E−04	418	47	−	4.45E−05	34	431	+	1.73E−06	0	465	+
CF_{10}	1.92E−06	1	464	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{11}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{12}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{13}	1.73E−06	0	465	+	2.89E−01	181	284	=	9.27E−03	106	359	+	7.34E−01	249	216	=
CF_{14}	1.73E−06	0	465	+	2.18E−02	344	121	−	2.07E−02	120	345	+	1.24E−05	20	445	+
CF_{15}	5.22E−06	11	454	+	1.36E−04	47	418	+	3.33E−02	129	336	+	1.73E−06	0	465	+
CF_{16}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{17}	5.22E−06	11	454	+	3.18E−06	459	6	−	4.73E−06	10	455	+	6.03E−02	99	366	+
CF_{18}	3.85E−03	92	373	+	2.06E−01	294	171	=	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{19}	6.89E−05	39	426	+	5.44E−01	262	203	=	0.4652	268	197	=	7.50E−01	217	248	=
CF_{20}	1.25E−04	46	419	+	4.39E−03	94	371	+	1.49E−05	443	22	−	7.19E−02	320	145	=
CF_{21}	7.71E−04	69	396	+	4.11E−03	372	93	−	6.34E−06	13	452	+	1.40E−02	352	113	−
CF_{22}	6.84E−03	101	364	+	4.07E−05	432	33	−	7.73E−03	362	103	−	2.77E−03	378	87	−
CF_{23}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{24}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{25}	1.73E−06	0	465	+	2.50E−01	0	465	=	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{26}	1.48E−02	351	114	−	6.16E−04	399	66	−	2.61E−04	410	55	−	2.84E−05	436	29	−
CF_{27}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{28}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{29}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{30}	1.73E−06	0	465	+	1.48E−03	78	387	+	1.73E−06	0	465	+	1.73E−06	0	465	+

described in Table 14 [76–78]. A constraint-handling approach needs to be implemented owing to various constraints in different problems. There are multiple methods which can deal with constraint problems, among which the penalty function method is the simplest and is utilized in this experiment [79,80]. Typically, the constrained engineering optimization problems of minimization are expressed as follows:

$$\text{Minimize} \quad f(\vec{x}), \vec{x} \in R^d \tag{16}$$

$$\text{Subject to} \begin{cases} g_i(\vec{x}) \le 0 & i = 1, \ldots, p \\ h_j(\vec{x}) = 0 & j = 1, \ldots, q \end{cases} \tag{17}$$

where g_i and h_i are the inequality and equality constraints, respectively, and R^d is an n-dimensional vector space over the field of real numbers. AHA needs to find the best feasible solution $\vec{x} = \{x_1, \ldots, x_d\}$ which minimizes the objective function $f(\vec{x})$ subject to the constraints.

Therefore, the constrained engineering optimization problems of minimization through the penalty function method are redefined as [79,81]:

$$\text{Minimize } F(\vec{x}) = \begin{cases} f(\vec{x}) & \vec{x} \in S \\ f(\vec{x}) + \lambda(\sum_{i=1}^{p} g_i^2(\vec{x}) + \sum_{j=1}^{q} h_j^2(\vec{x})) & \vec{x} \notin S \end{cases} \tag{18}$$

where S is the feasible search space. According to Eq. (18), the function will be assigned a big function value when the solution violates any constraint with any level, hence, the algorithm will automatically remove the infeasible solutions during the optimization process. The engineering cases are described mathematically in Appendix C.

Table 10
Wilcoxon signed rank test results for GSA, ABC, CMA-ES and SHADE versus AHA.

Fun.	GSA vs AHA				ABC vs AHA				CMA-ES vs AHA				SHADE vs AHA			
	p-value	T+	T−	Winner	*p*-value	T+	T−	Winner	*p*-value	T+	T−	Winner	*p*-value	T+	T−	Winner
CF_1	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.02E−05	447	18	−
CF_2	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	465	0	−
CF_3	1.73E−06	0	465	+	1.73E−06	0	465	+	1.92E−06	1	464	+	1.73E−06	465	0	−
CF_4	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	8.47E−06	449	16	−
CF_5	1.73E−06	465	0	−	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_6	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	4.07E−02	133	332	+
CF_7	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	465	0	−
CF_8	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_9	9.27E−03	106	359	+	1.73E−06	0	465	+	7.71E−04	69	396	+	5.72E−01	205	260	=
CF_{10}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.92E−06	1	464	+	1.73E−06	0	465	+
CF_{11}	1.92E−06	1	464	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{12}	1.73E−06	465	0	−	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{13}	1.73E−06	0	465	+	1.24E−05	20	445	+	1.73E−06	0	465	+	3.06E−04	408	57	−
CF_{14}	1.73E−06	0	465	+	4.90E−04	63	402	+	1.73E−06	0	465	+	1.41E−01	304	161	=
CF_{15}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.24E−05	20	445	+	1.78E−01	298	167	=
CF_{16}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{17}	1.73E−06	0	465	+	1.92E−06	1	464	+	2.60E−06	4	461	+	3.52E−06	458	7	−
CF_{18}	1.56E−02	350	115	−	3.32E−04	58	407	+	1.25E−02	111	354	+	1.49E−05	22	443	+
CF_{19}	1.73E−06	0	465	+	6.14E−01	257	208	=	2.83E−04	56	409	+	1.73E−06	465	0	−
CF_{20}	1.73E−06	0	465	+	1.92E−06	1	464	+	2.05E−04	52	413	+	1.73E−06	465	0	−
CF_{21}	1.73E−06	0	465	+	3.88E−06	8	457	+	3.61E−03	91	374	+	7.69E−06	450	15	−
CF_{22}	1.92E−06	1	464	+	3.49E−01	187	278	=	4.99E−03	96	369	+	5.31E−05	429	36	−
CF_{23}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{24}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{25}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{26}	2.41E−04	54	411	+	1.48E−03	387	78	−	6.16E−04	399	66	−	2.13E−06	463	2	−
CF_{27}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{28}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{29}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{30}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+	2.41E−04	54	411	+

4.3.1. Three-bar truss design

The task of this case study is to minimize the weight of a statically loaded three-bar truss and it has three constraints [82]. From the structure of this truss and its forces in Fig. 12, two parameters need to be determined to adjust the sectional areas.

AHA is used for this case and runs 30 times with 50 hummingbirds and 15,000 evaluations. The results offered by AHA are compared with those offered by these reported optimizers, such as SC [82], PSO-DE [83], DEDS [11], HEAA [84], and CS [85] in previous studies. Table 15 provides the comparison results between AHA and the other methods. As observed in this table, although PSO-DE and AHA achieve the best results in terms of the 'Worst', 'Mean' and 'Best' metrics, AHA requires less FEs. In addition, with the same FEs, AHA exhibits a significant advantage over DEDS, HEAA, and CS. The values of the objective function, variables and constraints, and number of FEs for this best design obtained by AHA are given in Table 16.

4.3.2. Cantilever beam design

For this well-known case [86], Fig. 13 shows the shape of the cantilever beam, which is supported at the leftmost block and the other blocks are left free. The beam's fixed end has a vertical force. Thus, the objective of this design is to minimize the weight of the beam. This case has five decision variables representing the lengths of different blocks and one constraint.

Some previous algorithms, including SOS [87], CS [85], MMA [86] GCA-I [86], GCA-II [86], MFO [88], and PSO-DE, were applied to this case. The results of AHA are compared with those of the above approaches in Table 17. From this table, although the 'Mean' metric of AHA is slightly inferior to that of SOS, while in terms of the 'Best' metric, AHA finds the better solution. Besides, AHA performs better than PSO-DE in terms of the

Table 11
Wilcoxon signed rank test results for WOA, SSA and BOA versus AHA.

Fun.	WOA vs AHA				SSA vs AHA				BOA vs AHA			
	p-value	T+	T−	Winner	*p*-value	T+	T−	Winner	*p*-value	T+	T−	Winner
CF_1	1.73E−06	0	465	+	3.60E−01	188	277	=	1.73E−06	0	465	+
CF_2	1.73E−06	0	465	+	1.73E−06	465	0	−	1.73E−06	0	465	+
CF_3	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_4	1.73E−06	0	465	+	4.17E−01	272	193	=	1.73E−06	0	465	+
CF_5	1.73E−06	0	465	+	1.73E−06	465	0	−	1.73E−06	0	465	+
CF_6	1.73E−06	0	465	+	4.45E−05	34	431	+	1.73E−06	0	465	+
CF_7	1.73E−06	0	465	+	1.73E−06	465	0	−	1.73E−06	0	465	+
CF_8	1.73E−06	0	465	+	1.92E−06	1	464	+	1.73E−06	0	465	+
CF_9	1.73E−06	0	465	+	7.19E−01	250	215	=	1.73E−06	0	465	+
CF_{10}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{11}	1.73E−06	0	465	+	1.24E−05	20	445	+	1.73E−06	0	465	+
CF_{12}	1.73E−06	0	465	+	1.13E−05	19	446	+	1.73E−06	0	465	+
CF_{13}	8.22E−03	104	361	+	8.59E−02	149	316	=	1.73E−06	0	465	+
CF_{14}	8.73E−03	105	360	+	2.06E−01	171	294	=	1.73E−06	0	465	+
CF_{15}	1.73E−06	0	465	+	4.68E−03	370	95	−	1.73E−06	0	465	+
CF_{16}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{17}	1.73E−06	0	465	+	7.66E−01	218	247	=	4.73E−06	10	455	+
CF_{18}	1.73E−06	0	465	+	2.41E−04	54	411	+	1.73E−06	0	465	+
CF_{19}	2.35E−06	3	462	+	9.92E−01	232	233	=	1.73E−06	0	465	+
CF_{20}	1.73E−06	0	465	+	1.29E−03	76	389	+	2.35E−06	3	462	+
CF_{21}	1.73E−06	0	465	+	9.75E−01	234	231	=	1.73E−06	0	465	+
CF_{22}	1.48E−04	48	417	+	7.27E−03	363	102	−	1.73E−06	0	465	+
CF_{23}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{24}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{25}	4.38E−04	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{26}	7.52E−02	319	146	=	1.49E−05	443	22	−	7.81E−01	246	219	=
CF_{27}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{28}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{29}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+
CF_{30}	1.73E−06	0	465	+	1.73E−06	0	465	+	1.73E−06	0	465	+

Table 12
Statistical results of Wilcoxon signed-rank test.

Function characteristics	PSO vs AHA (+/=/−)	TLBO vs AHA (+/=/−)	DE vs AHA (+/=/−)	CS vs AHA (+/=/−)	GSA vs AHA (+/=/−)	ABC vs AHA (+/=/−)
Unimodal	3/0/0	1/0/2	1/0/2	3/0/0	3/0/0	3/0/0
Multimodal	13/0/0	7/2/4	12/0/1	12/1/0	11/0/2	13/0/0
Hybrid	6/0/0	1/2/3	3/1/2	2/2/2	5/0/1	4/2/0
Composition	7/0/1	6/1/1	7/0/1	7/0/1	8/0/0	7/0/1
Total	29/0/1	15/5/10	23/1/6	24/3/3	27/0/3	27/2/1
Function characteristics	CMA-ES vs AHA (+/=/−)	SHADE vs AHA (+/=/−)	WOA vs AHA (+/=/−)	SSA vs AHA (+/=/−)	BOA vs AHA (+/=/−)	
Unimodal	3/0/0	0/0/3	3/0/0	1/1/1	3/0/0	
Multimodal	13/0/0	7/3/3	13/0/0	6/4/3	13/0/0	
Hybrid	6/0/0	1/0/5	6/0/0	2/3/1	6/0/0	
Composition	7/0/1	7/0/1	7/1/0	7/0/1	7/1/0	
Total	29/0/1	15/3/12	29/1/0	16/8/6	29/1/0	

'Mean' metric with the same FEs. It can be seen that AHA achieves the high-quality solution for this case. The comparative results show that our method can effectively solve this case and reveal better design.

Table 13
Performance rank of each algorithm for each function.

Fun.	AHA	PSO	TLBO	DE	CS	GSA	ABC	CMA-ES	SHADE	WOA	SSA	BOA
CF_1	3	9	1	6	5	12	10	7	2	8	4	11
CF_2	5	10	4	3	8	11	6	9	1	7	2	12
CF_3	3	5	7	2	4	11	10	6	1	12	8	9
CF_4	4	10	3	5	7	11	6	9	1	8	2	12
CF_5	3	9	12	5	7	1	10	6	8	4	2	11
CF_6	2	6	1	7	10	8.5	8.5	5	3	12	4	11
CF_7	5	9	4	3	8	11	6	10	1	7	2	12
CF_8	1	4.5	2	3	9	8	10	6	4.5	11	7	12
CF_9	2.5	8	1	6	9	5	10	7	4	11	2.5	12
CF_{10}	1	2	9	4	7	6	11	3.5	8	10	5	12
CF_{11}	1	8	10	7	4	3	11	9	6	5	2	12
CF_{12}	2	8	11	5	4	1	10	9	7	6	3	12
CF_{13}	3	9	1	5	2	11	7	10	6	8	4	12
CF_{14}	4	10	2	5	7	11	1	9	3	8	6	12
CF_{15}	3	6	5	4	9	11	7	8	2	10	1	12
CF_{16}	1	7	4	3	10	12	8	6	5	11	2	9
CF_{17}	3	10	2	8	5	12	6	9	1	11	4	7
CF_{18}	2	10	1	7	11	6	5	9	4	8	3	12
CF_{19}	6.5	8.5	3.5	4	6.5	12	3.5	8.5	1	10	3.5	11
CF_{20}	4	7	5	2	3	12	9	8	1	11	6	10
CF_{21}	4	6	2	7	3	12	8.5	8.5	1	11	5	10
CF_{22}	6	8	1	4	5	12	7	9	2	10	3	11
CF_{23}	1	10	5	6	8	3	7	11	4	12	9	2
CF_{24}	1	10	3	7	12	5	11	8	6	4	9	2
CF_{25}	1	11	3	10	7	4	12	9	5	8	6	2
CF_{26}	11	7	8	3	1	12	6	5	4	9	2	10
CF_{27}	1	10	5	7	4	12	9	8	2	11	3	6
CF_{28}	1	11	3	5	4	8	6	12	2	9	7	10
CF_{29}	1	11	2	5	7	9	6	12	3	10	4	8
CF_{30}	1	9	2	4	5	12	6	8	3	10	7	11

Table 14
Characteristics of 10 constrained engineering cases. NV: number of variables, NCV: number of continuous variables, NDV: number of discrete variables, NC: number of constraints, NIC: number of inequality constraints, NAC: number of active constraints, F/S: ratio between the feasible solutions in the search space (F) and the entire search space (S), DO: design objective.

No.	Case name	NV	NCV	NDV	NC	NIC	NAC	F/S	DO
1	Three-bar truss	2	2	0	3	3	–	–	Minimum weight
2	Cantilever beam	5	5	0	1	1	–	–	Minimum weight
3	Tension/compression spring	3	3	0	4	4	2	0.1	Minimum weight
4	Rolling element bearing	10	9	1	9	9	4	0.015	Maximum weight
5	Belleville spring	4	3	1	7	7	4	0.004	Minimum weight
6	Hydrostatic thrust bearing	4	4	0	7	7	3	0.003	Minimum power loss
7	Pressure vessel	4	2	2	4	4	2	0.40	Minimum cost
8	Welded beam	4	4	0	7	7	2	0.035	Minimum cost
9	Speed reducer	7	6	1	11	11	3	0.004	Minimum weight
10	Multiple disc clutch brake	5	0	5	8	8	1	0.700	Minimum mass

4.3.3. Tension/compression spring design

The intention of this engineering case [87] is to minimize the weight of a spring to satisfy four constraints with three geometric parameters. Fig. 14 describes the geometric structure of the spring. This case has three geometric variables and four constraints considered.

The results of AHA versus those previous techniques, including GA2 [89], GA3 [90], CA [91], CPSO [92], HPSO [93], PSO2 [94], QPSO [94], UPSO [95], CDE [96], SSB [82], PSO-DE [83], and $(\mu+\lambda)$ES [97], are listed in Table 18 to make a clear comparison. It can be observed that with less computational burden, AHA is superior to

Table 15
Results of different methods for three-bar truss design.

Methods	Worst	Mean	Best	Std	FEs
SC	263.969756	263.903356	263.895846	1.3E−02	17,610
PSO-DE	263.895843	263.895843	263.895843	1.2E−10	17,600
DEDS	263.895849	263.895843	263.895843	9.7E−07	15,000
HEAA	263.896099	263.895865	263.895843	4.9E−05	15,000
CS	–	264.0669	263.97156	9.0E−05	15,000
AHA	263.895843	263.895843	263.895843	1.09E−07	15,000

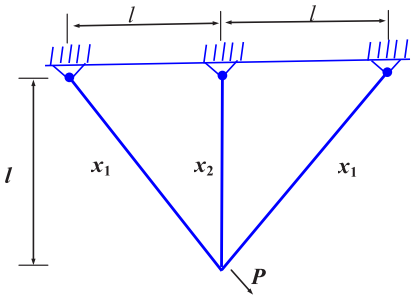

Fig. 12. Three-bar truss design problem.

Table 16
Best results offered by AHA for three-bar truss design, cantilever beam design, tension/compression spring design, rolling element bearing design, as well as Belleville spring design cases.

	Three-bar truss design	Cantilever beam design	Tension/compression spring design	Rolling element bearing design	Belleville spring design
$f(x)$	263.895843	1.3399650	0.0126660	85547.49822	1.979675
x_1	0.788683	6.01380	0.051897	125.718411	0.20414
x_2	0.4082246	5.302425	0.361748	21.425350	0.200000
x_3	–	4.496347	10.689283	10.527979	10.030467
x_4	–	3.508429	–	0.515000	12.009995
x_5	–	2.152705	–	0.515155	–
x_6	–	–	–	0.470216	–
x_7	–	–	–	0.640818	–
x_8	–	–	–	0.300012	–
x_9	–	–	–	0.095122	–
x_{10}	–	–	–	0.682241	–
g_1	−2.4166E−09	−6.2594E−06	−4.8700E−08	3.893956	0.003332
g_2	−1.464128	–	−7.9213E−09	9.935546	1.0848E−4
g_3	−0.535871	–	−4.063608	2.006616	2.7588E−08
g_4	–	–	−0.724235	24.498935	1.595856
g_5	–	–	–	23.062112	4.4988E−06
g_6	–	–	–	0.718411	1.979528
g_7	–	–	–	2.54117E−4	0.198965
g_8	–	–	–	−0.958104	–
g_9	–	–	–	2.8565E−09	–
g_{10}	–	–	–	1.5475E−4	–
FEs	15,000	15,000	25,000	15,000	24,000

others for finding the best optimal results in terms of the ‘Mean’ and ‘Best’ metrics. From Table 18, with less FEs, AHA provides the similar results as those of PSO-DE in terms of the ‘Worst’, ‘Mean’ and ‘Best’ metrics. Even in terms of the ‘Worst’ metric, AHA is superior to 10 out of 12 algorithms. These satisfactory results assure us that AHA can offer the best geometric variables with the minimum weight of the spring compared with its competitors.

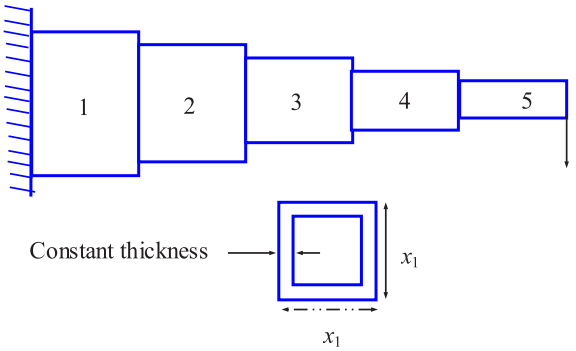

Fig. 13. Cantilever beam design problem.

Table 17
Results of different methods for cantilever beam design.

Methods	Worst	Mean	Best	Std	FEs
SOS	–	1.33997	1.33996	1.1E−05	15,000
CS	–	–	1.33999	–	125,000
MMA	–	–	1.3400	–	–
GCA-I	–	–	1.3400	–	–
GCA-II	–	–	1.3400	–	–
MFO	–	–	3.399880	–	–
PSO-DE	1.340191	1.340219	1.339957	6.61E−05	15,000
AHA	1.343036	1.340146	1.339957	7.912E−05	15,000

Table 18
Results of different methods for tension/compression spring design.

Methods	Worst	Mean	Best	Std	FEs
GA2	0.0128221	0.0127690	0.0127047	3.9390 E-05	900,000
GA3	0.0129730	0.0127420	0.0126810	5.9000 E-05	80,000
CA	0.0151156	0.0135681	0.0127210	8.4215 E-04	50,000
CPSO	0.0129240	0.0127300	0.0126747	5.1985 E-04	200,000
HPSO	0.0127191	0.0127072	0.0126652	1.5824 E-05	75,000
PSO2	0.0718020	0.0195550	0.0128570	0.0116620	2000
QPSO	0.0181270	0.0138540	0.0126690	1.3410E−03	2000
UPSO	0.0503651	0.0229478	0.0131200	7.2057E−03	10,000
CDE	0.0127900	0.0127030	0.0126702	2.7000E−05	240,000
SSB	0.0167173	0.0129227	0.0126692	5.9000E−04	25,167
(μ+λ)ES	–	0.0131650	0.0126890	3.9000E−04	30,000
PSO-DE	0.0126652	0.0126652	0.0126652	4.9E−09	42,100
AHA	0.0127271	0.0126976	0.0126660	1.5660E−05	25,000

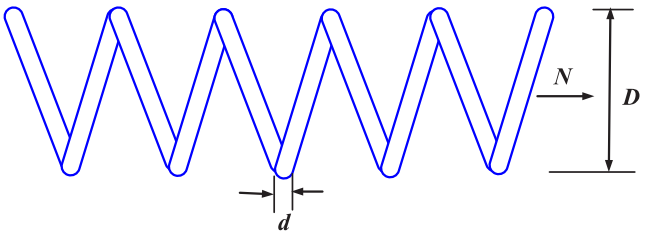

Fig. 14. Tension/compression spring design.

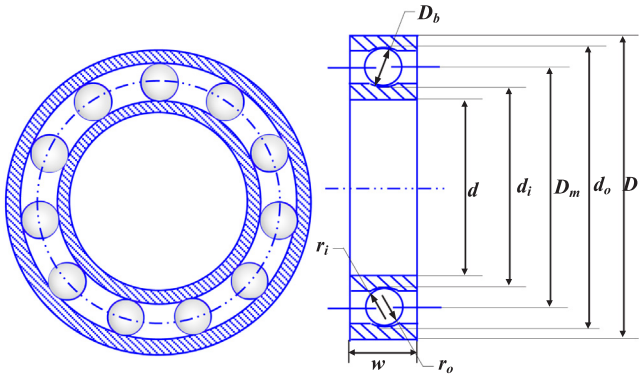

Fig. 15. Rolling element bearing design.

Table 19
Results of different methods for rolling element bearing design.

Methods	Worst	Mean	Best	Std	FEs
GA4	–	–	81843.3000	–	225,000
ABC	78897.8100	81496.0000	81859.7416	–	10,000
TLBO	80807.8551	81438.9870	81859.7400	–	10,000
MBA	84440.1948	85321.4030	85535.9611	211.5200	15,100
PVS	78897.81	80803.57	81859.59	–	10,000
PVS	79834.79	81550	81859.74	–	20,000
MDDE	81701.18	81848.7	81858.83	–	10,000
HHO	–	–	83011.88329	–	15,000
PSO-DE	85411.62816	84511.63285	85521.6172	2641.62	15,000
AHA	77385.6122	84635.8129	85547.49822	2111.211	15,000

4.3.4. Rolling element bearing design

For this study [98,99] depicted in Fig. 15, the load-carrying capacity of the bearing needs to be maximized. This case includes ten constraints and ten geometric variables.

This case was solved by a number of works in the literature such as GA4 [98], TLBO [100], ABC [100], MBA [101], PVS [77], MDDE [102], PSO-DE, and HHO [78]. The comparisons between the results of AHA and the other considered methods are given in Table 19. This engineering case is more difficult because of so many variables and constraints, as well as a low rate of the feasible solution space to the entire search space. From Table 19, most of the compared methods cannot provide desirable results in terms of the 'Best' metric with acceptable number of FEs. With the same number of evaluations, our method can provide the best geometric variables with the maximum load-carrying capacity for this case. Although AHA is inferior to PSO-DE in terms of the 'Worst' metric, AHA is superior to PSO-DE in terms of the 'Mean', 'Best' and 'Std' metrics.

4.3.5. Belleville spring design

This case requires to minimize the weight of the spring with seven constraints and its structure is depicted in Fig. 16 [103]. As this figure shows, we need to tackle four decision variables regarding this design.

The case was solved by some meta-heuristic methods, including GA5 [103], GeneAS I [104], ABC, TLBO, PSO-DE, and GeneAS II [104]. The results of these methods are compared to those of AHA in Table 20. As seen in this table, AHA provides the same best results as TLBO in terms of the 'Best' metric. But in terms of the 'Worst', 'Mean', and 'Std' metrics, our algorithm displays its superiority over all the other methods with the same number of FEs for this case. Remarkably, our method is superior to other algorithms in terms of solution quality and robustness.

Table 20
Results of different methods for Belleville spring design.

Methods	Worst	Mean	Best	Std	FEs
ABC	2.291062	2.2134251	1.988257	0.0452	24,000
TLBO	2.1149819	1.9886513	1.979675	0.0267	24,000
GA5	–	–	2.121964	–	24,000
GeneAS I	–	–	2.01807	–	24,000
GeneAS II	–	–	2.16256	–	24,000
PSO-DE	2.1134295	2.0028367	1.9882629	0.0539	24000
AHA	2.1041923	1.9860209	1.979675	0.0228	24,000

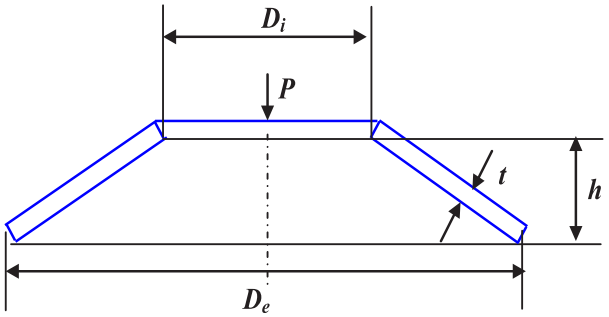

Fig. 16. Belleville spring design problem.

Table 21
Results of different methods for hydrostatic thrust bearing design.

Methods	Worst	Mean	Best	Std	FEs
IPSO	–	1757.376840	1632.2149	16.8510	90,000
GASO	–	–	1950.2860	–	16,000
TLBO	2096.80127	1797.7079	1625.443	–	50,000
ABC	2144.8360	1861.5540	1625.443	–	50,000
Gene AS	–	–	2161.6	–	–
BGA	–	–	2295.1	–	–
PSO-DE	2196.8219	1952.8137	1636.6281	216.32	50,000
AHA	1850.3812	1680.7812	1625. 4498	57	50,000

4.3.6. Hydrostatic thrust bearing design

The case requires minimizing the power loss by considering seven constraints [105]. Fig. 17 depicts the cross-section of the hydrostatic thrust bearing with four variables to be optimized.

This case was tackled by AHA using 50 hummingbirds with the number of evaluations at 50,000. This case was also solved by the previous methods, including IPSO [106], GASO [103], TLBO [100], ABC [100], GeneAS [104], PSO-DE, and BGA [104]. The results of these methods are compared with those of AHA in Table 21. Inspecting this table, AHA outperforms PSO-DE in terms of the 'Worst', 'Mean', 'Best' and 'Std' metrics with the same FEs; the 'Best' metric obtained by AHA matches those obtained by ABC and TLBO. However, in terms of the 'Mean' and 'Worst' metrics, our method is more successful than all the comparative counterparts. The values of the objective function, variables, and constraints and number of FEs for this best design obtained so far are given in Table 22.

4.3.7. Pressure vessel design

The objective is to minimize the fabrication cost of a pressure vessel, which is depicted in Fig. 18 [107]. The case includes four design variables and four constraints.

This case study was previously tackled by many heuristic techniques, including GA2 [89], GA3 [90], CPSO [92], HPSO [93], PSO-DE [83], PSO2 [94], CDE [96], ABC [100], QPSO [94], $(\mu + \lambda)$ES [97], and CSA [49]. AHA

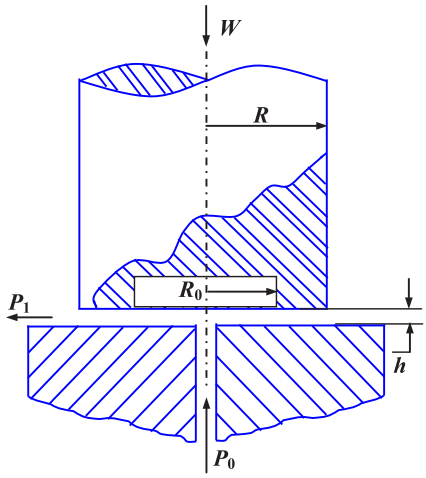

Fig. 17. Hydrostatic thrust bearing design.

Table 22
Best results offered by AHA for hydrostatic thrust bearing design, pressure vessel design, multiple disc clutch brake design, welded beam design, and speed reducer design.

	Hydrostatic thrust bearing design	Pressure vessel design	Welded beam design	Speed reducer design	Multiple disc clutch brake design
$f(x)$	1625. 4498	58853.5369	1.724853	2994.471158	0.3136566
x_1	5.955782	0.778171	0.205730	3.500000	70
x_2	5.389014	0.384653	3.470492	0.7000000	90
x_3	5.358745	40.319674	9.036624	17.000000	1
x_4	2.269694	199.999262	0.205730	7.300001	840
x_5	–	–	–	7.7153201	3
x_6	–	–	–	3.350212	–
x_7	–	–	–	5.286655	–
x_8	–	–	–	–	–
x_9	–	–	–	–	–
x_{10}	–	–	–	–	–
g_1	0.041272	−1.0887E−06	−2.5843E−03	−0.073915	0
g_2	1.8107E−4	−3.6397E−06	−3.5352E−03	−0.197999	24
g_3	7.6288E−4	−0.162464	−1.27535E−07	−0.499172	0.916444
g_4	3.2437E−4	−40.000737	−3.432983	−0.904644	9.824089
g_5	0.566768	–	−0.080730	−2.4087E−08	7.894696
g_6	9.96361E−4	–	−0.235540	−1.0358E−09	1.197971
g_7	4.6461E−3	–	−1.8003E−05	−0.702500	41.32500
g_8	–	–	–	−2.4311E−08	13.802029
g_9	–	–	–	−0.795833	–
g_{10}	–	–	–	−0.051326	–
g_{11}	–	–	–	−1.7760E−08	–
FEs	50,000	30,000	30,000	30,000	600

is applied to this case study with the number of evaluations at 30,000. The statistical results provided by these employed methods are depicted in Table 23. AHA provides a better solution than the other considered optimizers in terms of the 'Worst', 'Mean' and 'Best' metrics. This result shows that our optimizer can obtain the solution with the minimum fabrication cost under less computational efforts.

4.3.8. Welded beam design

The aim of this case is to find the minimum cost of a welded beam to satisfy two constraints with four control variables. The structure of the beam is depicted in Fig. 19.

Table 23
Results of different methods for pressure vessel design.

Methods	Worst	Mean	Best	Std	FEs
GA2	6308.4970	6293.8432	6288.7445	7.4133	900,000
GA3	6469.3220	6177.2533	6059.9463	130.9297	80,000
CPSO	6363.8041	6147.1332	6061.0777	86.4500	240,000
HPSO	6288.6770	6099.9323	6059.7143	86.2000	81,000
PSO-DE	6059.7143	6059.7143	6059.7143	1.0E−10	42,100
PSO2	14076.3240	8756.6803	6693.7212	1492.5670	8000
QPSO	8017.2816	6440.3786	6059.7209	479.2671	8000
CDE	6371.0455	6085.2303	6059.7340	43.0130	204,800
ABC	–	6245.3080	6059.714339	205.0000	30,000
(μ+λ)ES	6820.397461	6379.938037	6059.701610	210.0000	30,000
CSA	7332.8410	6342.4990	6059.7140	384.9450	250,000
AHA	5885.85190	5885.53823	5885.35369	0.1378	30,000

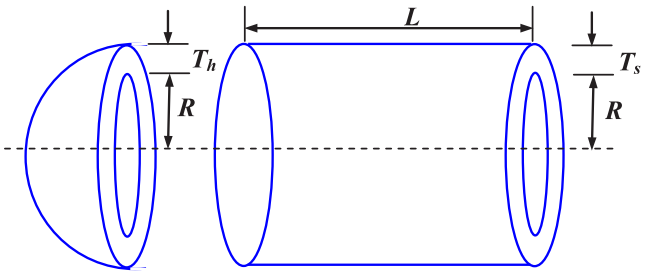

Fig. 18. Pressure vessel design problem.

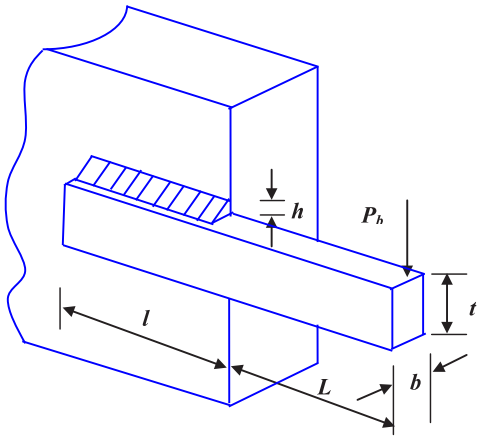

Fig. 19. Welded beam design problem.

This case is solved by AHA and the reported methods, including GA2 [84], GA3 [85], CPSO [92], HPSO [93], CDE [96], NM-PSO [108], HS [109], EPSO [110], CAEP [91], ABC [100], WCA [111], $(\mu + \lambda)$ES [97], PSO-DE [83], and SC [82]. The results provided by different methods are listed in Table 24. AHA performs more successfully than those reported methods with a lower number of FEs. Although the best best-so-far solutions provided by NM-PSO is slightly superior to AHA, for NM-PSO to reach this solution requires two times the number of evaluations larger than that AHA requires. Moreover, it can be seen that AHA using 30,000 FEs provides better performance than PSO-DE using 33,000 FEs in terms of the 'Worst', 'Mean' and 'Best' metrics. These results imply that our method can effectively search and find the optimal geometric variables of the welded beam.

Table 24
Results of different methods for welded beam design.

Methods	Worst	Mean	Best	Std	FEs
GA2	1.785835	1.771973	1.748309	1.1200E−02	900,000
GA3	1.993408	1.792654	1.728226	7.4700E−02	80,000
CPSO	1.782143	1.748831	1.728024	1.2900E−02	240,000
HPSO	1.814295	1.749040	1.724852	4.0100E−02	81,000
CDE	1.824105	1.768158	1.733462	2.2194E−02	204,800
NM-PSO	1.733393	1.726373	1.724717	3.50E−03	80,000
EPSO	1.7472200	1.728219	1.724853	5.6200E−03	50,000
HS	–	–	2.38	–	110,000
CAEP	3.179709	1.971809	1.724852	0.443131	50,000
ABC	–	1.741913	1.724852	3.100E−02	30,000
WCA	1.801127	1.735940	1.724857	1.89E−02	30,000
(μ+λ)ES	–	1.777692	1.724852	8.800E−02	30,000
SC	6.3996780	3.0025883	2.3854347	9.600E−01	33,095
PSO-DE	1.7248811	1.7248579	1.7248531	4.1E−06	33,000
AHA	1.7248528	1.7248524	1.7248523	1.34E−7	30,000

Table 25
Comparisons of statistical results using reported optimizers in literature for speed reducer design.

Methods	Worst	Mean	Best	Std	FEs
SC	3009.9647360	3001.7582640	2994.7442410	–	54,456
HEAA	2994.7523110	2994.6133680	2994.4991070	7.0E−02	40,000
ABC	–	–	2997.058412	–	30,000
PSO-DE	2996.348166	2996.348165	2996.348165	1.0E−07	70,100
PVS	2996.348165	2996.348165	2996.348165	–	54,350
FFA	2996.669	2996.51	2996.37	–	50,000
CSA	30090	3007.1997	3000.98	–	5,000
(μ+λ)ES	–	2996.3480940	2996.3480940	0	30,000
MDE	–	2996.3672200	2996.3566890	8.2E−03	24,000
AHA	2994.473229	2994.471652	2994.471158	4.2512E−4	30,000

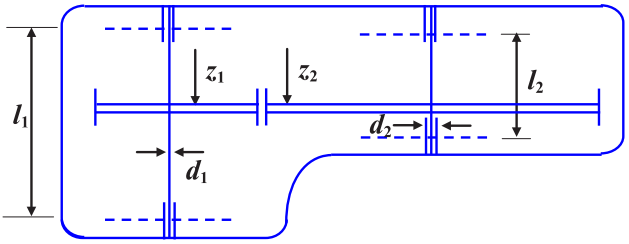

Fig. 20. Speed reducer design.

4.3.9. Speed reducer design

The task in this case is to design a speed reducer to minimize its weigh [97]. The six geometric variables of the speed reducer are described in Fig. 20, and obtaining the minimum weight requires satisfying 11 constraints.

This case was previously tackled by many scholars using various heuristic methods, including SC [82], HEAA [84], ABC [100], PSO-DE [83], $(\mu + \lambda)$ES [97], MDE [112], PVS [77], FFA [76] and CSA [49]. The results of these methods are compared to those of AHA. The comparisons of the results obtained by these various optimizers including AHA are shown in Table 25 for this case. As it can be seen in Table 25, among the algorithms using more than or equal to 30,000 evaluations, AHA performs the best with a lower number of evaluations in terms of the 'Worst', 'Best' and 'Mean' metrics. It also can be found that although AHA employs more evaluations than MDE, our method can find better geometric variables for this case.

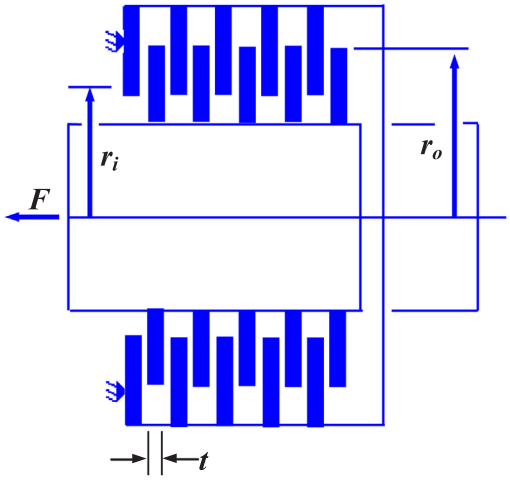

Fig. 21. Multiple disc clutch brake design.

Table 26
Comparisons of statistical results using reported optimizers in literature for multiple disc clutch brake design.

Methods	Worst	Mean	Best	Std	FEs
ABC	0.352864	0.324751	0.313657	0.54	600
TLBO	0.392071	0.3271662	0.313657	0.67	600
PVS	0.352864	0.333652	0.313657	–	600
PSO-DE	0.3379216	0.3236821	0.313657	0.0091	600
AHA	0.3332601	0.3216842	0.3136566	0.0076	600

4.3.10. Multiple disc clutch brake design

This engineering case requires designing a multiple disc clutch brake for the minimum mass [113]. Fig. 21 depicts the structure of such a brake and the discrete geotropic variables. This design has five related discrete variables and eight complex constraints.

The case was handled using the reported heuristic techniques involving ABC [100], TLBO [100], PSO-DE, and PVS [77]. Our method is also run to search for the global optimal solution of the case with the same number of FEs. Table 26 summarizes the obtained results of different methods. In terms of different metrics, AHA detects the best results with the same computational burden compared to the other methods. The convincing results provide the evidence of AHA's competitiveness in dealing with this case.

The results and analyses of Section 4 strongly show the merits of the proposed AHA algorithm in solving optimization problems. In the next section, a very challenging, real-world problem in the area of hydropower operation design will be used to further demonstrate the efficiency and applicability of this algorithm.

5. Hydropower operation design using AHA

Hydropower, as a renewable and clean energy source, has accounted for a considerably large proportion in the electric power, which is not able to meet the growing demands of people owing to social and economic development [114,115]. Therefore, generating more power by hydropower plants and enhancing their utility are becoming increasingly important. The operations of reservoir and hydropower systems are very complicated due to the uncertainty of inflow and other factors, so how to perform the optimal operation of a hydropower plant is a task of top priority. Fig. 22 shows the sketch of hydropower operation [116]. We optimize the operation of a hydropower plant to maximize its gross power generation in a scheduling period, in which the optimal water levels

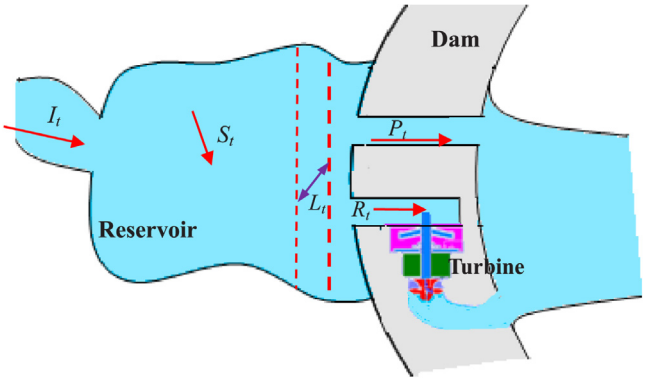

Fig. 22. Sketch of hydropower operation.

can be determined. Thus, the hydropower operation design problem can be considered as a maximization problem with constraints as follows [117,118]:

$$\text{Consider variable } \vec{L} = [L_1, L_2, L_3, L_4, L_5, L_6, L_7, L_8, L_9, L_{10}, L_{11}, L_{12}] \tag{19}$$

$$\text{Maximize } E(\vec{L}) = \sum_{t=1}^{T=12} a R_t H_t M_t \tag{20}$$

$$\text{Variable range } L_{tmin} \le L_t \le L_{tmax}, t = 1, \ldots, 12 \tag{21}$$

where L_t is the water level in the tth time interval, E is the gross energy of a hydropower plant over T months, a is the power efficiency of hydropower units, R_t is the generating flow in the tth time interval, H_t is the average head in the tth time interval, M_t is the time length in the tth time interval, and L_{tmin} and L_{tmax} are the minimal and maximal water levels in the tth time interval, respectively. The constraints are as follows.

(1) Water balance constraint:

$$S_{t+1} = S_t + I_t - R_t - P_t \tag{22}$$

where S_t is the storage in the tth time interval, I_t is the inflow in the tth time interval, R_t is the power discharge in the tth time interval, and P_t is the spill in the tth time interval.

(2) Reservoir storage capacity constraint:

$$S_{tmin} \le S_t \le S_{tmax} \tag{23}$$

where S_{tmin} and S_{tmax} are the minimal and maximal operational storages in the tth time interval, respectively.

(3) Power discharge constraint:

$$R_{tmin} \le R_t \le R_{tmax} \tag{24}$$

where R_{tmin} and R_{tmax} are the minimal and maximal power discharges in the tth time interval, respectively.

(4) Hydropower plant power constraint:

$$E_{tmin} \le a R_t H_t \le E_{tmax} \tag{25}$$

where E_{tmin} and E_{tmax} are the minimal and maximal energy generations in the tth time interval, respectively. The relationship between water level and the storage is:

$$S_t = U(L_t) \tag{26}$$

Three tests are implemented: the low flow year, the medium flow year, and the high flow year. Each test employs 30 individuals over 9000 FEs. Figs. 23–25 provide the convergence curves and the obtained optimal head levels over 12 months in each test, respectively. The maximal obtained gross power generations of the low water year, medium water year, and high water year are 4.458×10^9 kW·h, 4.787×10^9 kW·h, and 5.107×10^9 kW·h, respectively.

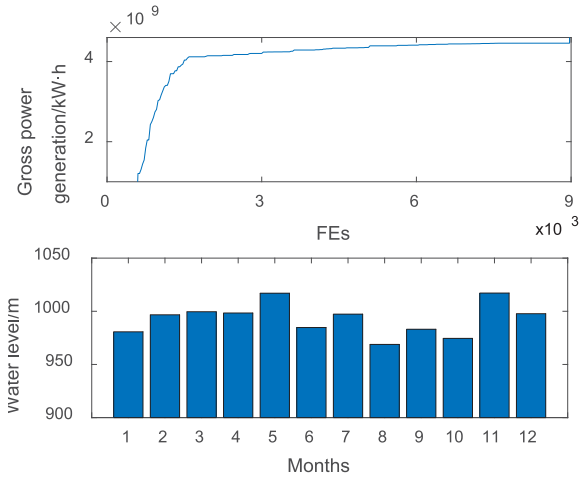

Fig. 23. Convergence curve of low water year and its water levels over 12 months.

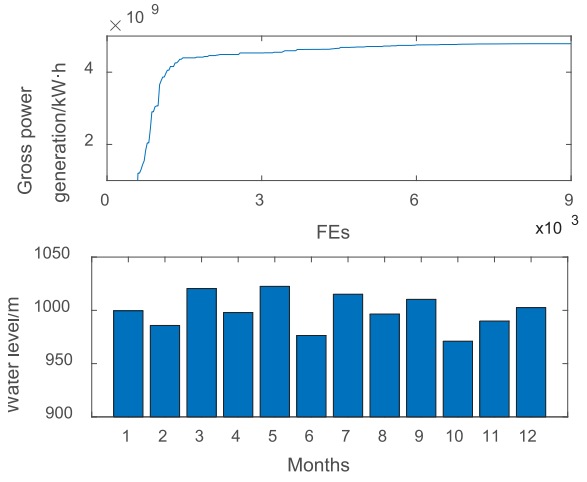

Fig. 24. Convergence curve of medium water year and its water levels over 12 months.

Observing these figures, it can be found that AHA can easily detect the optimal water level of every month which maximizes the gross power generation and shows a good convergence rate. Additionally, from these figures, it can be observed that the infeasible water levels are obtained in the initial stage which make the gross power generation unreasonable, while the gross power generation increases as the iterations proceed and the best feasible water levels are found in the last iteration.

It should be noted that there is an obvious difference among the optimal water levels offered by AHA for the same months in different flow years. This is because the inflows in the upstream are different during the same period in different flow years, causing the change of the discharge and capacity of reservoir in the objective functions and constraints. This change generally forces the optimal water levels to shift from a position to another according to the flow years with various flow levels. These results on the hydropower operation design suggest that our algorithm

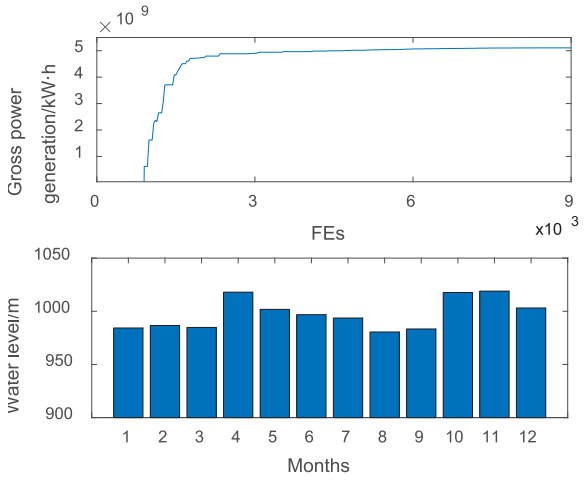

Fig. 25. Convergence curve of high water year and its water levels over 12 months.

is able to effectively offer the optimal water levels to improve the generation efficiency of the hydropower plant. It further reveals the potential of AHA in tackling the challenging real-world problems with unknown and constrained variable space.

6. Conclusions

A novel bio-inspired optimization technique called the AHA algorithm, inspired from the flight skills and foraging behaviors of hummingbirds, is proposed for dealing with different optimization tasks in this work. Three foraging behaviors, along with the memory function for food sources, and three flight patterns, are jointly simulated for solving global optimization. Two sets of different benchmark functions are employed to investigate the optimization performance of AHA, the results obtained demonstrate that AHA can find the global optimum compared to other well-regarded algorithms. Additionally, ten well-known engineering cases studies are employed to further check the performance of AHA. The results display its success and competitiveness in terms of computational burden and solution precision. Eventually, AHA is employed to tackle the hydropower operation design, the experimental results justify the applicability and potential of the proposed optimizer in practice. In designing a standard version of AHA, we try to keep it simple. Thus, there are several research directions to follow to improve the algorithm in future work. This original version can be promoted using various strategies. On the one hand, some stochastic operators can be integrated into AHA to develop an improved version, including useful heuristics, such as deterministic operators, chaotic maps, and local searching behaviors. On the other hand, AHA may be equipped with some search components from other optimization techniques to design a hybridized version. Besides, other variants of AHA can be developed to tackle binary or multi-objective problems.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors sincerely thank the anonymous reviewers for their valuable and insightful comments and suggestions. The authors would like to thank Dr. Jason of University of Illinois at Urbana-Champaign for editorial review. This work was supported in part by the National Natural Science Foundation of China (11972144 and 12072098), and the One Hundred Outstanding Innovative Scholars of Colleges and Universities in Hebei Province of China (SLRC2019022).

Appendix A. 50 benchmark functions used in experiment 1

A.1. Benchmark functions, D: Dimensions, C: Characteristics, U: Unimodal, M: Multimodal, S: Separable, N: Non-separable

Name	C	Function	D	Range	f_{opt}
Stepint	US	$f_1(x) = 25 + \sum_{i=1}^{n} \lfloor x_i \rfloor$	5	$[-5.12, 5.12]$	0
Step	US	$f_2(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	30	$[-100, 100]$	0
Sphere	US	$f_3(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
SumSquares	US	$f_4(x) = \sum_{i=1}^{n} i x_i^2$	30	$[-10, 10]$	0
Quartic	US	$f_5(x) = \sum_{i=1}^{n} i x_i^4 + random\,[0, 1)$	30	$[-1.28, 1.28]$	0
Beale	UN	$f_6(x) = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2$	5	$[-4.5, 4.5]$	0
Easom	UN	$f_7(x) = -\cos(x_1)\cos(x_2)\mathrm{e}^{-(x_1-\pi)^2-(x_2-\pi)^2}$	2	$[-100, 100]$	-1
Matyas	UN	$f_8(x) = 0.26(x_1^2 + x_2^2) - 0.48x^1x^2$	2	$[-10, 10]$	0
Colville	UN	$f_9(x) = 100(x_1 - x_2)^2 + (x_1 - 1)^2 + (x_4 - 1)^2 + 90(x_3^2 - x^4)^2 + 10.1((x_2 - 1)^2 + (x_4 - 1)^2) + 19.8(x_2 - 1)(x_4 - 1)$	4	$[-10, 10]$	0
Trid6	UN	$f_{10}(x) = \sum_{i=1}^{n} (x_i - 1)^2 + \sum_{i=2}^{n} x_i x_{i-1}$	6	$[-D^2, D^2]$	-50
Trid10	UN	$f_{11}(x) = \sum_{i=1}^{n} (x_i - 1)^2 + \sum_{i=2}^{n} x_i x_{i-1}$	10	$[-D^2, D^2]$	-210
Zakharov	UN	$f_{12}(x) = \sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} 0.5 i x_i)^2 + (\sum_{i=1}^{n} 0.5 i x_i)^4$	10	$[-5, 10]$	0
Powell	UN	$f_{13}(x) = \sum_{i=1}^{n/k} (x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + 5(x_{4i-2} - x_{4i-1})^4 + 10(x_{4i-1} - x_{4i})^4$	24	$[-4, 5]$	0
Schwefel 2.22	UN	$f_{14}(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
Schwefel 1.2	UN	$f_{15}(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	30	$[-100, 100]$	0
Rosenbrock	UN	$f_{16}(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i)^2) + (x_i - 1)^2$	30	$[-30, 30]$	0
Dixon–Price	UN	$f_{17}(x) = (x_1 - 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2$	30	$[-10, 10]$	0
Foxholes	MS	$f_{18}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{j=1}^{2} (x_i - a_{ij})^6}\right]^{-1}$	2	$[-65.536, 65.536]$	0.998
Branin	MS	$f_{19}(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos x_1 + 10$	2	$[-5, 10] \times [0, 15]$	0.398
Bohachevsky1	MS	$f_{20}(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$	2	$[-100, 100]$	0

A.2. Benchmark functions, D: Dimensions, C: Characteristics, U: Unimodal, M: Multimodal, S: Separable, N: Non-separable

Name	C	Function	D	Range	f_{opt}
Booth	MS	$f_{21}(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	2	$[-10, 10]$	0
Rastrigin	MS	$f_{22}(x) = -\sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]$	0
Schwefel	MS	$f_{23}(x) = -\sum_{i=1}^{n} (x_i \sin(\sqrt{\lvert x_i \rvert}))$	30	$[-500, 500]$	-12569.5
Michalewicz2	MS	$f_{24}(x) = -\sum_{i=1}^{n} \sin(x_i)(\sin(i x_i^2/\pi))^{20}$	2	$[0, \pi]$	-1.8013
Michalewicz5	MS	$f_{25}(x) = -\sum_{i=1}^{n} \sin(x_i)(\sin(i x_i^2/\pi))^{20}$	5	$[0, \pi]$	-4.6877
Michalewicz10	MS	$f_{26}(x) = -\sum_{i=1}^{n} \sin(x_i)(\sin(i x_i^2/\pi))^{20}$	10	$[0, \pi]$	-9.6602
Schaffer	MN	$f_{27}(x) = 0.5 + \frac{\sin^2(\sqrt{x_1^2 + x_2^2}) - 0.5}{(1 + 0.001(x_1^2 + x_2^2))^2}$	2	$[-100,100]$	0
Six Hump Camel Back	MN	$f_{28}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5,5]$	-1.03163
Bohachevsky2	MN	$f_{29}(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1)(4\pi x_3) + 0.3$	2	$[-100,100]$	0
Bohachevsky3	MN	$f_{30}(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1 + 4\pi x_3) + 0.3$	2	$[-100,100]$	0

Shubert	MN	$f_{31}(x) = (\sum_{i=1}^{5} i\cos((i+1)x_1 + i))$ $(\sum_{i=1}^{5} i\cos((i+1)x_2 + i))$	2	[−10,10]	-186.7309
GoldStein–Price	MN	$f_{32}(x) = \left[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right]$ $\times \left[30 + (2x_1 + 1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$	2	[−2,2]	3
Kowalik	MN	$f_{33}(x) = \sum_{i=1}^{11} \left\lvert a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right\rvert^2$	4	[−5,5]	0.00031
Shekel5	MN	$f_{34}(x) = -\sum_{i=1}^{5} \left\lvert (x_i - a_i)(x_i - a_i)^T + c_i \right\rvert^{-1}$	4	[0,10]	−10.1532
Shekel7	MN	$f_{35}(x) = -\sum_{i=1}^{7} \left\lvert (x_i - a_i)(x_i - a_i)^T + c_i \right\rvert^{-1}$	4	[0,10]	−10.4028
Shekel10	MN	$f_{36}(x) = -\sum_{i=1}^{10} \left\lvert (x_i - a_i)(x_i - a_i)^T + c_i \right\rvert^{-1}$	4	[0,10]	−10.5363
Perm	MN	$f_{37}(x) = \sum_{k=1}^{n} (\sum_{i=1}^{n} (i^k + \beta)((x_i/i)^k - 1))^2$	4	[−D,D]	0
PowerSum	MN	$f_{38}(x) = \sum_{k=1}^{n} ((\sum_{i=1}^{n} x_i^k) - b_k)^2$	4	[0,D]	0
Hartman3	MN	$f_{39}(x) = -\sum_{i=1}^{4} \exp\left[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right]$	3	[0,1]	−3.86
Hartman6	MN	$f_{40}(x) = -\sum_{i=1}^{4} \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right]$	6	[0,1]	−3.32

A.3. Benchmark functions, D: Dimensions, C: Characteristics, U: Unimodal, M: Multimodal, S: Separable, N: Non-separable

Name	C	Function	D	Range	f_{opt}
Griewank	MN	$f_{41}(x) = \frac{1}{4000}\sum_{i=1}^{n} (x_i - 100)^2$ $-\prod_{i=1}^{n} \cos(\frac{x_i - 100}{\sqrt{i}}) + 1$	30	[−600,600]	0
Ackley	MN	$f_{42}(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2})$ $-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i) + 20 + e$	30	[−32,32]	0
Penalized	MN	$f_{43}(x) = \frac{\pi}{n}\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2$ $\times [1 + 10\sin^2(\pi y_i + 1)]$ $+(y_n - 1)^2\} + \sum_{i=1}^{30} u(x_i, 10, 100, 4)$	30	[−50,50]	0
Penalized2	MN	$f_{44}(x) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{29}(x_i - 1)^2 p$ $[1 + \sin^2(3\pi x_{i+1})]$ $+(x_n - 1)^2[1 + \sin^2(2\pi x_{30})]\}$ $+\sum_{i=1}^{30} u(x_i, 5, 100, 4)$	30	[−50,50]	0
Langerman2	MN	$f_{45}(x) = -c_i(\exp(-\frac{1}{\pi}\sum_{j=1}^{n}(x_j - a_{ij})^2)$ $\times\ \cos(\pi\sum_{j=1}^{n}(x_j - a_{ij})^2))$	2	[0,10]	1.08
Langerman5	MN	$f_{46}(x) = -c_i(\exp(-\frac{1}{\pi}\sum_{j=1}^{n}(x_j - a_{ij})^2)$ $\times \cos(\pi\sum_{j=1}^{n}(x_j - a_{ij})^2))$	5	[0,10]	1.5
Langerman10	MN	$f_{47}(x) = -c_i(\exp(-\frac{1}{\pi}\sum_{j=1}^{n}(x_j - a_{ij})^2)$ $\times \cos(\pi\sum_{j=1}^{n}(x_j - a_{ij})^2))$	10	[0,10]	–
FletcherPowell2	MN	$f_{48}(x) = \sum_{i=1}^{n}(A_i - B_i)^2$ $A_i = \sum_{j=1}^{n}(a_{ij}\sin\alpha_j + b_{ij}\cos\alpha_j)$ $B_i = \sum_{j=1}^{n}(a_{ij}\sin x_j + b_{ij}\cos x_j)$	2	$[-\pi, \pi]$	0
FletcherPowell5	MN	$f_{49}(x) = \sum_{i=1}^{n}(A_i - B_i)^2$ $A_i = \sum_{j=1}^{n}(a_{ij}\sin\alpha_j + b_{ij}\cos\alpha_j)$ $B_i = \sum_{j=1}^{n}(a_{ij}\sin x_j + b_{ij}\cos x_j)$	5	$[-\pi, \pi]$	0
FletcherPowell0	MN	$f_{50}(x) = \sum_{i=1}^{n}(A_i - B_i)^2$ $A_i = \sum_{j=1}^{n}(a_{ij}\sin\alpha_j + b_{ij}\cos\alpha_j)$ $B_i = \sum_{j=1}^{n}(a_{ij}\sin x_j + b_{ij}\cos x_j)$	10	$[-\pi, \pi]$	0

Appendix B. CEC 2014 test suite used in experiment 2

B.1. CEC 2014 test functions

Name	Function	D	Range	f_{opt}
CF_1	Rotated High Conditioned Elliptic Function	30	[−100,100]	100
CF_2	Rotated Bent Cigar Function	30	[−100,100]	200
CF_3	Rotated Discus Function	30	[−100,100]	300
CF_4	Shifted and Rotated Rosenbrock's Function	30	[−100,100]	400
CF_5	Shifted and Rotated Ackley's Function	30	[−100,100]	500
CF_6	Shifted and Rotated Weierstrass Function	30	[−100,100]	600
CF_7	Shifted and Rotated Griewank's Function	30	[−100,100]	700
CF_8	Shifted Rastrigin's Function	30	[−100,100]	800
CF_9	Shifted and Rotated Rastrigin's Function	30	[−100,100]	900
CF_{10}	Shifted Schwefel's Function	30	[−100,100]	1000
CF_{11}	Shifted and Rotated Schwefel's Function	30	[−100,100]	1100
CF_{12}	Shifted and Rotated Katsuura Function	30	[−100,100]	1200
CF_{13}	Shifted and Rotated HappyCat Function	30	[−100,100]	1300
CF_{14}	Shifted and Rotated HGBat Function	30	[−100,100]	1400
CF_{15}	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	30	[−100,100]	1500
CF_{16}	Shifted and Rotated Expanded Scaffer's F6 Function	30	[−100,100]	1600
CF_{17}	Hybrid Function 1 (N=3)	30	[−100,100]	1700
CF_{18}	Hybrid Function 2 (N=3)	30	[−100,100]	1800
CF_{19}	Hybrid Function 3 (N=4)	30	[−100,100]	1900
CF_{20}	Hybrid Function 4 (N=4)	30	[−100,100]	2000
CF_{21}	Hybrid Function 5 (N=5)	30	[−100,100]	2100
CF_{22}	Hybrid Function 6 (N=5)	30	[−100,100]	2200
CF_{23}	Composition Function 1 (N=5)	30	[−100,100]	2300
CF_{24}	Composition Function 2 (N=3)	30	[−100,100]	2400
CF_{25}	Composition Function 3 (N=3)	30	[−100,100]	2500
CF_{26}	Composition Function 4 (N=5)	30	[−100,100]	2600
CF_{27}	Composition Function 5 (p=5)	30	[−100,100]	2700
CF_{28}	Composition Function 6 (N=5)	30	[−100,100]	2800
CF_{29}	Composition Function 7 (N=3)	30	[−100,100]	2900
CF_{30}	Composition Function 8 (N=3)	30	[−100,100]	3000

Appendix C. 10 engineering design cases in experiment 3

C.1. Three-bar truss design

Consider variable $\vec{x} = [x_1, x_2]$.
Minimize $f_1(\vec{x}) = (2\sqrt{2}x_2 + x_2) \times l$.
Subject to $g_1(\vec{x}) = \frac{\sqrt{2}x_1+x_2}{\sqrt{2}x_1^2+2x_2x_1} P - \sigma \leq 0$, $g_2(\vec{x}) = \frac{x_2}{\sqrt{2}x_1^2+2x_2x_1} P - \sigma \leq 0 \leq 0$, $g_3(\vec{x}) = \frac{x_2}{x_1+\sqrt{2}x_2} P - \sigma \leq 0$.
Where l = 10 cm, P = 2 KN/cm^2, σ = 2 KN/cm^2.
Variable range $0 \leq x_1 \leq 1$, $0 \leq x_2 \leq 1$.

C.2. Cantilever beam design

Consider variable $\vec{x} = [x_1, x_2]$.
Minimize $f_2(\vec{x}) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5)$.

Table C.27
Variation of $f(a)$ with a.

a	≤ 1.4	1.5	1.6	1.7	1.8	1.9	2	2.1	2.2	2.3	2.4	2.5	2.6	2.7	≥ 2.8
$f(a)$	1	0.85	0.77	0.71	0.66	0.63	0.6	0.58	0.56	0.55	0.53	0.52	0.51	0.51	0.5

Subject to $g_1(\vec{x}) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \leq 0$.
Variable range $0.01 \leq x_i \leq 100$, $i = 1, \ldots, 5$.

C.3. Tension/compression spring design

Consider variable $\vec{x} = [x_1, x_2, x_3] = [d, D, N]$.
Minimize $f_3(\vec{x}) = (x_3 + 2)x_2x_1^2$.
Subject to $g_1(\vec{x}) = 1 - \frac{x_3x_2^3}{71785x_1^4} \leq 0$, $g_2(\vec{x}) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \leq 0$, $g_3(\vec{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0$, $g_4(\vec{x}) = \frac{x_1+x_2}{1.5} - 1 \leq 0$.
Variable range $0.05 \leq x_1 \leq 2$, $0.25 \leq x_2 \leq 1.3$, $2 \leq x_3 \leq 15$

C.4. Rolling element bearing design

Consider variable $\vec{x} = [D_m, D_b, Z, f_i, f_o, K_{Dmin}, K_{Dmax}, \varepsilon, e, \zeta]$.
Maximize $\begin{cases} f_4(\vec{x}) = f_cZ^{2/3}D_b^{1.8} \; if \; D_b \leq 25.4 \text{ mm} \\ f_4(\vec{x}) = 3.647 f_cZ^{2/3}D_b^{1.4} \; if \; D_b > 25.4 \text{ mm} \end{cases}$.
Subject to $g_1(\vec{x}) = \frac{\phi_o}{2\sin^{-1}(D_b/D_m)} - Z + 1 \geq 0$, $g_2(\vec{x}) = 2D_b - K_{Dmin}(D-d) \geq 0$, $g_3(\vec{x}) = K_{Dmax}(D-d) - 2D_b \geq 0$, $g_4(\vec{x}) = D_m - (0.5 - e)(D + d) \geq 0$, $g_5(\vec{x}) = (0.5 + e)(D + d) - D_m \geq 0$, $g_6(\vec{x}) = D_m - 0.5(D + d) \geq 0$, $g_7(\vec{x}) = 0.5(D - D_m - D_b) - \varepsilon D_b \geq 0$, $g_8(\vec{x}) = \zeta B_w - D_b \leq 0$, $g_9(\vec{x}) = f_i \geq 0.515$, $g_{10}(\vec{x}) = f_o \geq 0.515$.
Where $f_c = 37.91\left[1 + \left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\left(\frac{f_i(2f_o-1)}{f_o(2f_i-1)}\right)^{0.4}\right\}^{10/3}\right]^{-0.3}\left(\frac{\gamma^{0.3}(1-\gamma)^{1.39}}{f_o(1+\gamma)^{\frac{1}{3}}}\right)\left(\frac{2f_i}{2f_i-1}\right)^{0.41}$, $\gamma = \frac{D_b\cos\alpha}{D_m}$, $f_i = \frac{r_i}{D_b}$, $\phi_o = 2\pi - 2\cos^{-1}\frac{\{(D-d)/2-3(T/4)\}^2+\{D/2-(T/4)-D_b\}^2-\{d/2+(T/4)\}^2}{2\{(D-d)/2-3(T/4)\}\{D/2-(T/4)-D_b\}}$, $T = D - d - 2D_b$, $D = 160$, $d = 90$, $B_w = 30$, $r_i = r_o = 11.033$.
Variable range $0.5(D + d) \leq D_m \leq 0.6(D + d)$, $0.15(D - d) \leq D_b \leq 0.45(D - d)$, $4 \leq Z \leq 50$, $0.515 \leq f_i \leq 0.6$, $0.515 \leq f_o \leq 0.6$, $0.4 \leq K_{Dmin} \leq 0.5$, $0.6 \leq K_{Dmax} \leq 0.7$, $0.3 \leq \varepsilon \leq 0.4$, $0.02 \leq e \leq 0.1$, $0.6 \leq \zeta \leq 0.85$.

C.5. Belleville spring design

Consider variable $\vec{x} = [t, h, D_i, D_e]$.
Minimize $f_5(\vec{x}) = 0.07075\pi\left(D_e^2 - D_i^2\right)t$.
Subject to $g_1(\vec{x}) = S - \frac{4E\delta_{max}}{(1-\mu^2)\alpha D_e^2}\left[\beta(h - \frac{\delta_{max}}{2}) + \gamma t\right] \geq 0$, $g_2(\vec{x}) = \left(\frac{4E\delta_{max}}{(1-\mu^2)\alpha D_e^2}\left[(h - \frac{\delta}{2})((h - \delta)t + t^3)\right]\right)_{\delta=\delta_{max}} - P_{max} \geq 0$, $g_3(\vec{x}) = \delta_1 - \delta_{max} \geq 0$, $g_4(\vec{x}) = H - h - t \geq 0$, $g_5(\vec{x}) = D_{max} - D_e \geq 0$, $g_6(\vec{x}) = D_e - D_i \geq 0$, $g_7(\vec{x}) = 0.3 - \frac{h}{D_e-D_i} \geq 0$.
Where $\alpha = \frac{6}{\pi \ln K}\left(\frac{K-1}{K}\right)^2$, $\beta = \frac{6}{\pi \ln K}\left(\frac{K-1}{\ln K} - 1\right)$, $\gamma = \frac{6}{\pi \ln K}\left(\frac{K-1}{2}\right)$, P_{max} = 5400 lb, $\delta_{\max}$= 0.2 in, S = 200000 Psi, $E = 30 \times 10^6$ psi, $\mu = 0.3$, H= 2 in, D_{max} = 12.01 in, $K = D_e/D_i$, $\delta_1 = f(a)h$, $a = h/t$.
Values of $f(a)$ vary as shown in Table C.27.
Variable range $0.01 \leq t \leq 6$, $0.05 \leq h \leq 0.5$, $5 \leq D_i \leq 15$, $5 \leq D_o \leq 15$.

C.6. Hydrostatic thrust bearing design

Consider variable $\vec{x} = [R, R_o, \mu, Q]$.
Minimize $f_6(\vec{x}) = \frac{QP_o}{0.7} + E_f$.

Subject to $g_1(\vec{x}) = W - W_s \geq 0$, $g_2(\vec{x}) = P_{max} - P_o \geq 0$, $g_3(\vec{x}) = \Delta T_{max} - \Delta T \geq 0$, $g_4(\vec{x}) = h - h_{min} \geq 0$, $g_5(\vec{x}) = R - R_o \geq 0$, $g_6(\vec{x}) = 0.001 - \frac{\gamma}{gP_o}(\frac{Q}{2\pi Rh}) \geq 0$, $g_7(\vec{x}) = 5000 - \frac{W}{\pi(R^2 - R_o^2)} \geq 0$.

Where $W = \frac{\pi P_o}{2}\frac{R^2 - R_o^2}{\ln(R/R_o)}$, $P_o = \frac{6\mu Q}{\pi h^3}\ln(R/R_o)$, $E_f = 9336Q\gamma C\Delta T$, $\Delta T = 2(10^P - 560)$, $P = \frac{\log_{10}\log_{10}(8.122\times 10^6\mu + 0.8) - C_1}{n}$, $h = (\frac{2\pi N}{60})^2\frac{2\pi\mu}{E_f}(\frac{R^4}{4} - \frac{R_o^4}{4})$, $\gamma = 0.0307$, $C = 0.5$, $n = -3.55$, $C_1 = 10.04$, $W_s = 101000$, $P_{max} = 1000$, $h_{min} = 0.001$, $\Delta T_{max} = 50$, $g = 386.4$, $N = 750$.

Variable range $1 \leq R \leq 16$, $1 \leq R_o \leq 16$, $10^{-6} \leq \mu \leq 16 \times 10^{-6}$, $1 \leq Q \leq 16$.

C.7. Pressure vessel design

Consider variable $\vec{x} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L]$.

Minimize $f_7(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$.

Subject to $g_1(\vec{x}) = -x_1 + 0.0193x_3 \leq 0$, $g_2(\vec{x}) = -x_2 + 0.00954x_3 \leq 0$, $g_3(\vec{x}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0$, $g_4(\vec{x}) = x_4 - 240 \leq 0$.

Variable range $0 \leq x_1 \leq 99$, $0 \leq x_2 \leq 99$, $10 \leq x_3 \leq 200$, $10 \leq x_4 \leq 200$.

C.8. Welded beam design

Consider variable $\vec{x} = [x_1, x_2, x_3, x_4] = [h, l, t, b]$.

Minimize $f_8(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2)$.

Subject to $g_1(\vec{x}) = \tau(\vec{x}) + \tau_{max} \leq 0$, $g_2(\vec{x}) = \sigma(\vec{x}) + \sigma_{max} \leq 0$, $g_3(\vec{x}) = \delta(\vec{x}) + \delta_{max} \leq 0$, $g_4(\vec{x}) = x_1 - x_4 \leq 0$, $g_5(\vec{x}) = P - P_c(\vec{x}) \leq 0$, $g_6(\vec{x}) = 0.125 - x_1 \leq 0$, $g_7(\vec{x}) = 0.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \leq 0$.

Where $\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$, $\tau' = \frac{P}{\sqrt{2}x_1x_2}$, $\tau'' = \frac{MR}{J}$, $M = P(L + \frac{x_2}{2})$, $R = \sqrt{\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2}$, $J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^3}{4} + (\frac{x_1 + x_3}{2})^2\right]\right\}$, $\sigma(\vec{x}) = \frac{6PL}{x_4x_3^2}$, $\delta(\vec{x}) = \frac{4PL^3}{Ex_4x_3^3}$, $P_c(\vec{x}) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$, $P = 6000$ lb, $L = 14$ in, $E = 30 \times 10^6$ psi, $G = 12 \times 10^6$ psi, $\tau_{max} = 13600$ psi, $\sigma_{max} = 30000$ psi, $\delta_{max} = 0.25$ in.

Variable range $0.1 \leq x_1 \leq 2$, $0.1 \leq x_2 \leq 10$, $0.1 \leq x_3 \leq 10$, $0.1 \leq x_4 \leq 2$.

C.9. Speed reducer design

Consider variable $\vec{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7] = [b, m, z, l_1, l_2, d_1, d_2]$.

Minimize $f_9(\vec{x}) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 0.7854x_1(x_4x_6^2 - x_5x_7^2)$.

Subject to $g_1(\vec{x}) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0$, $g_2(\vec{x}) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0$, $g_3(\vec{x}) = \frac{1.93x_4^2}{x_2x_6^4x_3} - 1 \leq 0$, $g_4(\vec{x}) = \frac{1.93x_5^2}{x_2x_7^4x_3} - 1 \leq 0$, $g_5(\vec{x}) = \frac{((\frac{745x_4}{x_2x_3})^2 + 16.9\times 10^6)^{0.5}}{110x_6^3} - 1 \leq 0$, $g_6(\vec{x}) = \frac{((\frac{745x_5}{x_2x_3})^2 + 157.5\times 10^6)^{0.5}}{85x_7^3} - 1 \leq 0$, $g_7(\vec{x}) = \frac{x_2x_3}{40} - 1 \leq 0$, $g_8(\vec{x}) = \frac{5x_2}{x_1} - 1 \leq 0$, $g_9(\vec{x}) = \frac{x_1}{12x_2} - 1 \leq 0$, $g_{10}(\vec{x}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0$, $g_{11}(\vec{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0$.

Variable range $2.6 \leq x_1 \leq 3.6$, $0.7 \leq x_2 \leq 0.8$, $17 \leq x_3 \leq 28$, $7.3 \leq x_4 \leq 8.3$, $7.3 \leq x_5 \leq 8.3$, $2.9 \leq x_6 \leq 3.9$, $5.0 \leq x_7 \leq 5.5$.

C.10. Multiple disc clutch brake design

Consider variable $\vec{x} = [r_i, r_o, t, F, Z]$.

Minimize $f_{10}(\vec{x}) = \pi(r_0^2 - r_i^2)t(Z + 1)\rho$.

Subject to $g_1(\vec{x}) = r_o - r_i - \Delta r \geq 0$, $g_2(\vec{x}) = l_{max} - (Z + 1)(t + \delta) \geq 0$, $g_3(\vec{x}) = p_{max} - p_{rz} \geq 0$, $g_4(\vec{x}) = p_{max}v_{srmax} - p_{rz}v_{sr} \geq 0$, $g_5(\vec{x}) = v_{srmax} - v_{sr} \geq 0$, $g_6(\vec{x}) = T_{max} - T \geq 0$, $g_7(\vec{x}) = M_h - sM_s \geq 0$, $g_8(\vec{x}) = T \geq 0$.

Where $M_h = \frac{2}{3}\mu FZ\frac{r_o^3 - r_i^3}{r_o^2 - r_i^2}$, $p_{rz} = \frac{F}{\pi(r_o^2 - r_i^2)}$, $v_{sr} = \frac{2\pi n(r_o^3 - r_i^3)}{90(r_o^2 - r_i^2)}$, $T = \frac{I_z\pi n}{30(M_h + M_f)}$, $\Delta r = 20$ mm, $l_{max} = 30$ mm, $v_{srmax} = 10$ m/s, $\delta = 0.5$, $s = 1.5$, $M_s = 40$ Nm, $M_f = 3$ Nm, $n = 250$ rpm, $p_{max} = 1$ MPa, $I_z = 55$ kg mm^2, $T_{max} = 15$ s, $F_{max} = 1000$ N, $r_{imin} = 60$ mm, $r_{imax} = 80$ mm, $r_{omin} = 90$ mm, $r_{omax} = 110$ mm, $t_{min} = 1$ mm, $t_{max} = 3$ mm, $F_{min} = 600$, $F_{max} = 1100$, $Z_{min} = 2$, $Z_{max} = 9$.

Appendix D. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.cma.2021.114194.

References

[1] Z. Beheshti, S. Shamsuddin, A review of population-based meta-heuristic algorithms, Int. J. Adv. Soft Comput. Appl. 5 (1) (2011) 1–35.

[2] C. Blum, A. Roli, Meta-heuristics in combinatorial optimization: Overview and conceptual comparison, ACM Comput. Surv. 35 (2003) 268–308.

[3] T. Mantere, J. Alander, Evolutionary software engineering, a review, Appl. Soft Comput. 5 (3) (2005) 315–331.

[4] S. Yang, X. Gu, Y. Liu, R. Hao, S. Li, A general multi-objective optimized wavelet filter and its applications in fault diagnosis of wheelset bearings, Mech. Syst. Signal Process. 145 (2020) 106914.

[5] W. Zhao, L. Wang, Z. Zhang, Atom search optimization and its application to solve a hydrogeologic parameter estimation problem, Knowl.-Based Syst. 163 (2019) 283–304.

[6] H. Wang, Z. Ren, X. Li, X. Chen, H. Jiang, Solving team making problem for crowdsourcing with hybrid metaheuristic algorithm, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion, 2018, pp. 318–319.

[7] W. Zhao, L. Wang, Z. Zhang, Supply-demand-based optimization: a novel economics-inspired algorithm for global optimization, IEEE Access 7 (2019) 73182–73206.

[8] E. Rodríguez-Esparza, L.A. Zanella-Calzada, D. Oliva, A.A. Heidari, D. Zaldivar, M. Pérez-Cisneros, L.K. Foong, An efficient Harris hawks-inspired image segmentation method, Expert Syst. Appl. 155 (2020) 113428.

[9] I.A. Doush, E. Santos, A sensitivity analysis for harmony search with multi-parent crossover algorithm, in: Proceedings of SAI Intelligent Systems Conference, Springer, 2019, pp. 276–284.

[10] N. Ferro, S. Micheletti, S. Perotto, An optimization algorithm for automatic structural design, Comput. Methods Appl. Mech. Engrg. 372 (2020) 113335.

[11] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, A.H. Gandomi, The arithmetic optimization algorithm, Comput. Methods Appl. Mech. Engrg. 376 (2021) 113609.

[12] W. Zhao, T. Shi, L. Wang, Q. Cao, H. Zhang, An adaptive hybrid atom search optimization with particle swarm optimization and its application to optimal no-load PID design of hydro-turbine governor, J. comput. Des. Eng. 8 (5) (2021) 1204–1233.

[13] W. Zhao, C. Du, S. Jiang, An adaptive multiscale approach for identifying multiple flaws based on XFEM and a discrete artificial fish swarm algorithm, Comput. Methods Appl. Mech. Engrg. 339 (2018) 341–357.

[14] A. Darwish, Bio-inspired computing: Algorithms review, deep analysis, and the scope of applications, Future Comput. Inform. J. 3 (2) (2018) 231–246.

[15] A. Hatamlou, A hybrid bio-inspired algorithm and its application, Appl. Intell. 47 (4) (2017) 1059–1067.

[16] J.H. Holland, Genetic algorithms, Sci. Am. 267 (1) (1992) 66–73.

[17] A. Gogna, A. Tayal, Metaheuristics: review and application, J. Exp. Theor. Artif. Intell. 25 (4) (2013) 503–526.

[18] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95-International Conference on Neural Networks, Vol. 4, IEEE, 1995, pp. 1942–1948.

[19] Y. Zhang, S. Wang, G. Ji, A comprehensive survey on particle swarm optimization algorithm and its applications, Math. Probl. Eng. 2015 (2015).

[20] M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybern. B 26 (1) (1996) 29–41.

[21] M.N. Ab Wahab, S. Nefti-Meziani, A. Atyabi, A comprehensive review of swarm optimization algorithms, PLoS One 10 (5) (2015) e0122827.

[22] D. Karaboga, B. Akay, A comparative study of artificial bee colony algorithm, Appl. Math. Comput. 214 (1) (2009) 108–132.

[23] G. Yan, C. Li, An effective refinement artificial bee colony optimization algorithm based on chaotic search and application for PID control tuning, J. Comput. Inf Syst. 7 (9) (2011) 3309–3316.

[24] X.S. Yang, S. Deb, Cuckoo search via Lévy flights, in: 2009 World Congress on Nature & Biologically Inspired Computing, NaBIC, IEEE, 2009, pp. 210–214.

[25] X.S. Yang, S. Deb, Engineering optimisation by cuckoo search, Int. J. Math. Model. Numer. Optim. 1 (4) (2010) 330–343.

[26] S. Yin, J. Liu, Y. Zhang, L. Teng, Cuckoo search algorithm based on mobile cloud model, Int. J. Innovative Comput. Inf. Control 12 (6) (2016) 1809–1819.

[27] J. Wang, B. Zhou, S. Zhou, An improved cuckoo search optimization algorithm for the problem of chaotic systems parameter estimation, Comput. Intell. Neurosci. (2016) 1–8.

[28] X.S. Yang, A.H. Gandomi, Bat algorithm: a novel approach for global engineering optimization, Eng. Comput. 29 (5) (2012) 464–483.

[29] K.N. Krishnanand, D. Ghose, Detection of multiple source locations using a glowworm metaphor with applications to collective robotics, in: Proceedings 2005 IEEE Swarm Intelligence Symposium, SIS 2005, IEEE, 2005, pp. 84–91.

[30] W.T. Pan, A new fruit fly optimization algorithm: taking the financial distress model as an example, Knowl.-Based Syst. 26 (2012) 69–74.

[31] K.M. Passino, Biomimicry of bacterial foraging for distributed optimization and control, IEEE Control Syst. Mag. 22 (3) (2002) 52–67.

[32] M. Jain, V. Singh, A. Rani, A novel nature-inspired algorithm for optimization: Squirrel search algorithm, Swarm Evol. Comput. 44 (2019) 148–175.
[33] W. Zhao, L. Wang, Z. Zhang, Artificial ecosystem-based optimization: a novel nature-inspired meta-heuristic algorithm, Neural Comput. Appl. 32 (13) (2020) 9383–9425.
[34] A. Kaveh, N. Farhoudi, A new optimization method: Dolphin echolocation, Adv. Eng. Softw. 59 (2013) 53–70.
[35] O. Abedinia, N. Amjady, A. Ghasemi, A new metaheuristic algorithm based on shark smell optimization, Complexity 21 (5) (2016) 97–116.
[36] S. Mirjalili, A. Lewis, The whale optimization algorithm, Adv. Eng. Softw. 95 (2016) 51–67.
[37] M.D. Li, H. Zhao, X.W. Weng, T. Han, A novel nature-inspired algorithm for optimization: Virus colony search, Adv. Eng. Softw. 92 (2016) 65–88.
[38] A. Cheraghalipour, M. Hajiaghaei-Keshteli, M.M. Paydar, Tree growth algorithm (TGA): A novel approach for solving optimization problems, Eng. Appl. Artif. Intell. 72 (2018) 393–414.
[39] G. Dhiman, V. Kumar, Emperor penguin optimizer: A bio-inspired algorithm for engineering problems, Knowl.-Based Syst. 159 (2018) 20–50.
[40] A.R. Mehrabian, C. Lucas, A novel numerical optimization algorithm inspired from weed colonization, Ecol. Inform. 1 (4) (2006) 355–366.
[41] S. Arora, S. Singh, Butterfly optimization algorithm: a novel approach for global optimization, Soft Comput. 23 (3) (2019) 715–734.
[42] G. Dhiman, V. Kumar, Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications, Adv. Eng. Softw. 114 (2017) 48–70.
[43] A.H. Gandomi, A.H. Alavi, Krill herd: A new bio-inspired optimization algorithm, Commun. Nonlinear Sci. Numer. Simul. 17 (12) (2012) 4831–4845.
[44] S.H.S. Moosavi, V.K. Bardsiri, Satin bowerbird optimizer: A new optimization algorithm to optimize ANFIS for software development effort estimation, Eng. Appl. Artif. Intell. 60 (2017) 1–15.
[45] F. Merrikh-Bayat, The runner-root algorithm: A metaheuristic for solving unimodal and multimodal optimization problems inspired by runners and roots of plants in nature, Appl. Soft Comput. 33 (2015) 292–303.
[46] A. Askarzadeh, Bird mating optimizer: an optimization algorithm inspired by bird mating strategies, Commun. Nonlinear Sci. Numer. Simul. 19 (4) (2014) 1213–1228.
[47] S. Saremi, S. Mirjalili, A. Lewis, Grasshopper optimisation algorithm: theory and application, Adv. Eng. Softw. 105 (2017) 30–47.
[48] X.S. Yang, Flower pollination algorithm for global optimization, in: International Conference on Unconventional Computing and Natural Computation, Springer, 2012, pp. 240–249.
[49] A. Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm, Comput. Struct. 169 (2016) 1–12.
[50] Z. Meng, J.-S. Pan, Monkey king evolution: A new memetic evolutionary algorithm and its application in vehicle fuel consumption optimization, Knowl.-Based Syst. 97 (2016) 144–157.
[51] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61.
[52] I. Fister Jr., X.-S. Yang, I. Fister, J. Brest, D. Fister, A brief review of nature-inspired algorithms for optimization, 2013, arXiv preprint arXiv:1307.4186.
[53] N.S. Jaddi, S. Abdullah, Optimization of neural network using kidney-inspired algorithm with control of filtration rate and chaotic map for real-world rainfall forecasting, Eng. Appl. Artif. Intell. 67 (2018) 246–259.
[54] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82.
[55] G. Zhu, S. Kwong, Gbest-guided artificial bee colony algorithm for numerical function optimization, Appl. Math. Comput. 217 (7) (2010) 3166–3173.
[56] Z. Yan, J. Zhang, J. Zeng, J. Tang, Nature-inspired approach: An enhanced whale optimization algorithm for global optimization, Math. Comput. Simulation 185 (2021) 17–46.
[57] S. Gupta, K. Deep, A memory-based grey wolf optimizer for global optimization tasks, Appl. Soft Comput. 93 (2020) 106367.
[58] Z. Zhang, C. Huang, D. Ding, S. Tang, B. Han, H. Huang, Hummingbirds optimization algorithm-based particle filter for maneuvering target tracking, Nonlinear Dynam. 97 (2) (2019) 1227–1243.
[59] B.A. Fennelly, Observations from the jewel rooms, Ecotone 8 (1) (2012) 74–85.
[60] D.L. Altshuler, R. Dudley, The ecological and evolutionary interface of hummingbird flight physiology, J. Exp. Biol. 205 (16) (2002) 2325–2336.
[61] B.J. Ward, L.B. Day, S.R. Wilkening, D.R. Wylie, D.M. Saucier, A.N. Iwaniuk, Hummingbirds have a greatly enlarged hippocampal formation, Biol. Lett. 8 (4) (2012) 657–659.
[62] M. Bateson, S.D. Healy, T.A. Hurly, Context–dependent foraging decisions in rufous hummingbirds, Proc. R. Soc. B 270 (1521) (2003) 1271–1276.
[63] J. Henderson, T.A. Hurly, M. Bateson, S.D. Healy, Timing in free-living rufous hummingbirds, *Selasphorus rufus*, Curr. Biol. 16 (5) (2006) 512–515.
[64] D. Griffiths, A. Dickinson, N. Clayton, Episodic memory: what can animals remember about their past? Trends Cogn. Sci. 3 (2) (1999) 74–80.
[65] D. Warrick, T. Hedrick, M.J. Fernandez, B. Tobalske, A. Biewener, Hummingbird flight, Curr. Biol. 22 (12) (2012) R472–R477.
[66] F. Leys, D. Reynaerts, D. Vandepitte, Outperforming hummingbirds' load-lifting capability with a lightweight hummingbird-like flapping-wing mechanism, Biol. Open 5 (8) (2016) 1052–1060.
[67] R. Storn, K. Price, Differential evolution–A simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.

[68] S.S. Reddy, B. Panigrahi, R. Kundu, R. Mukherjee, S. Debchoudhury, Energy and spinning reserve scheduling for a wind-thermal power system using CMA-ES with mean learning technique, Int. J. Electr. Power Energy Syst. 53 (2013) 113–122.
[69] R. Tanabe, A. Fukunaga, Success-history based parameter adaptation for differential evolution, in: 2013 IEEE Congress on Evolutionary Computation, IEEE, 2013, pp. 71–78.
[70] S. Mirjalili, A.H. Gandomi, S.Z. Mirjalili, S. Saremi, H. Faris, S.M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, Adv. Eng. Softw. 114 (2017) 163–191.
[71] H. Arsham, Global optima for linearly constrained business decision models, Sci. J. Adm. 2 (1) (2004) 27–53.
[72] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (1) (2011) 3–18.
[73] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, J. Amer. Statist. Assoc. 32 (200) (1937) 675–701.
[74] J. Hodges, E.L. Lehmann, Rank methods for combination of independent experiments in analysis of variance, Ann. Math. Stat. 33 (2) (1962) 482–497.
[75] J.J. Liang, B.Y. Qu, P.N. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization, Computational Intelligence Laboratory, Zhengzhou University Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, 635, IEEE New York, NY, 2013, p. 490.
[76] M. Fesanghary, M. Mahdavi, M. Minary-Jolandan, Y. Alizadeh, Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems, Comput. Methods Appl. Mech. Engrg. 197 (33–40) (2008) 3080–3091.
[77] P. Savsani, V. Savsani, Passing vehicle search (PVS): A novel metaheuristic algorithm, Appl. Math. Model. 40 (5–6) (2016) 3951–3978.
[78] A.A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future Gener. Comput. Syst. 97 (2019) 849–872.
[79] C.A.C. Coello, Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art, Comput. Methods Appl. Mech. Engrg. 191 (11–12) (2002) 1245–1287.
[80] L. Xia, L. Zhang, Q. Xia, T. Shi, Stress-based topology optimization using bi-directional evolutionary structural optimization method, Comput. Methods Appl. Mech. Engrg. 333 (2018) 356–370.
[81] W. Zhao, Z. Zhang, L. Wang, Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications, Eng. Appl. Artif. Intell. 87 (2020) 103300.
[82] T. Ray, K.M. Liew, Society and civilization: an optimization algorithm based on the simulation of social behavior, IEEE Trans. Evol. Comput. 7 (4) (2003) 386–396.
[83] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Appl. Soft Comput. 10 (2) (2010) 629–640.
[84] Y. Wang, Z. Cai, Y. Zhou, Z. Fan, Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique, Struct. Multidiscip. Optim. 37 (4) (2009) 395–413.
[85] A.H. Gandomi, X. Yang, A.H. Alavi, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, Eng. Comput. 29 (1) (2013) 17–35.
[86] H. Chickermane, H.C. Gea, Structural optimization using a new local approximation method, Internat. J. Numer. Methods Engrg. 39 (5) (1996) 829–846.
[87] M.Y. Cheng, D. Prayogo, Symbiotic organisms search: a new metaheuristic optimization algorithm, Comput. Struct. 139 (2014) 98–112.
[88] S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowl.-Based Syst. 89 (2015) 228–249.
[89] C.A.C. Coello, Use of a self-adaptive penalty approach for engineering optimization problems, Comput. Ind. 41 (2) (2000) 113–127.
[90] C.A.C. Coello, E.M. Montes, Constraint-handling in genetic algorithms through the use of dominance-based tournament selection, Adv. Eng. Inform. 16 (3) (2002) 193–203.
[91] C.A. Coello Coello, R.L. Becerra, Efficient evolutionary optimization through the use of a cultural algorithm, Eng. Optim. 36 (2) (2004) 219–236.
[92] Q. He, L. Wang, An effective co-evolutionary particle swarm optimization for constrained engineering design problems, Eng. Appl. Artif. Intell. 20 (1) (2007) 89–99.
[93] Q. He, L. Wang, A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization, Appl. Math. Comput. 186 (2) (2007) 1407–1422.
[94] L. dos Santos Coelho, Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems, Expert Syst. Appl. 37 (2) (2010) 1676–1683.
[95] K.E. Parsopoulos, M.N. Vrahatis, Unified particle swarm optimization for solving constrained engineering optimization problems, in: International Conference on Natural Computation, Springer, 2005, pp. 582–591.
[96] F.Z. Huang, L. Wang, Q. He, An effective co-evolutionary differential evolution for constrained optimization, Appl. Math. Comput. 186 (1) (2007) 340–356.
[97] E. Mezura-Montes, C.A.C. Coello, Useful infeasible solutions in engineering optimization with evolutionary algorithms, in: Mexican International Conference on Artificial Intelligence, Springer, 2005, pp. 652–662.
[98] B.R. Rao, R. Tiwari, Optimum design of rolling element bearings using genetic algorithms, Mech. Mach. Theory 42 (2) (2007) 233–250.
[99] A.A. Ewees, M. Abd Elaziz, Performance analysis of chaotic multi-verse harris hawks optimization: a case study on solving engineering problems, Eng. Appl. Artif. Intell. 88 (2020) 103370.
[100] R.V. Rao, V.J. Savsani, D. Vakharia, Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems, Comput. Aided Des. 43 (3) (2011) 303–315.

[101] A. Sadollah, A. Bahreininejad, H. Eskandar, M. Hamdi, Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems, Appl. Soft Comput. 13 (5) (2013) 2592–2612.
[102] W. Gong, Z. Cai, D. Liang, Engineering optimization by means of an improved constrained differential evolution, Comput. Methods Appl. Mech. Engrg. 268 (2014) 884–904.
[103] C.A.C. Coello, Treating constraints as objectives for single-objective evolutionary optimization, Eng. Optim. A35 32 (3) (2000) 275–308.
[104] K. Deb, M. Goyal, Optimizing engineering designs using a combined genetic search, in: ICGA, Citeseer, 1997, pp. 521–528.
[105] J.N. Siddall, Optimal Engineering Design: Principles and Applications, CRC Press, 1982.
[106] S. He, E. Prempain, Q. Wu, An improved particle swarm optimizer for mechanical design optimization problems, Eng. Optim. 36 (5) (2004) 585–605.
[107] B. Kannan, S.N. Kramer, An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design, J. Mech. Des. 116 (1994) 405–411.
[108] E. Zahara, Y. Kao, Hybrid Nelder–Mead simplex search and particle swarm optimization for constrained engineering design problems, Expert Syst. Appl. 36 (2) (2009) 3880–3886.
[109] K.S. Lee, Z.W. Geem, A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice, Comput. Methods Appl. Mech. Engrg. 194 (36–38) (2005) 3902–3933.
[110] T.T. Ngo, A. Sadollah, J.H. Kim, A cooperative particle swarm optimizer with stochastic movements for computationally expensive numerical optimization problems, J. Comput. Sci. 13 (2016) 68–82.
[111] H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, Water cycle algorithm–A novel metaheuristic optimization method for solving constrained engineering optimization problems, Comput. Struct. 110 (2012) 151–166.
[112] E. Mezura-Montes, C.C. Coello, J. Velázquez-Reyes, Increasing successful offspring and diversity in differential evolution for engineering design, in: Proceedings of the Seventh International Conference on Adaptive Computing in Design and Manufacture, ACDM 2006, 2006, pp. 131–139.
[113] A. Osyczka, Evolutionary Algorithms for Single and Multicriteria Design Optimization: Studies in Fuzzyness and Soft Computing, PhysicaVerlag, Heidelberg, 2002.
[114] M. Hossain, A. Huda, S. Mekhilef, M. Seyedmahmoudian, B. Horan, A. Stojcevski, M. Ahmed, A state-of-the-art review of hydropower in Malaysia as renewable energy: Current status and future prospects, Energy Strategy Rev. 22 (2018) 426–437.
[115] P. Murty, Chapter 24 - Renewable energy sources, in: P. Murty (Ed.), Electrical Power Systems, Butterworth-Heinemann, Boston, 2017, pp. 783–800.
[116] S. DeLand, Solving large-scale optimization problems with MATLAB: A hydroelectric flow example, 2012, Website https://www.mathworks.com/company/newsletters/articles/solving-large-scale-optimization-problems-with-matlab-a-hydroelectric-flow-example.html.
[117] P. Yong, G. Tang, Z. Xue, Optimal operation of cascade reservoirs based on improved artificial fish swarm algorithm, Syst. Eng. Theory Pract. 31 (6) (2011) 1118–1125.
[118] G. Zhou, R. Zhao, Q. Luo, Y. Zhou, Optimal hydropower station dispatch using quantum social spider optimization algorithm, Concurr. Comput.: Pract. Exper. (2020) e5782.