Applied Intelligence
https://doi.org/10.1007/s10489-020-01893-z

Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems

Fatma A. Hashim[1] · Kashif Hussain[2] · Essam H. Houssein[3] · Mai S. Mabrouk[4] · Walid Al-Atabany[1]

Abstract

The difficulty and complexity of the real-world numerical optimization problems has grown manifold, which demands efficient optimization methods. To date, various metaheuristic approaches have been introduced, but only a few have earned recognition in research community. In this paper, a new metaheuristic algorithm called Archimedes optimization algorithm (AOA) is introduced to solve the optimization problems. AOA is devised with inspirations from an interesting law of physics Archimedes' Principle. It imitates the principle of buoyant force exerted upward on an object, partially or fully immersed in fluid, is proportional to weight of the displaced fluid. To evaluate performance, the proposed AOA algorithm is tested on CEC'17 test suite and four engineering design problems. The solutions obtained with AOA have outperformed well-known state-of-the-art and recently introduced metaheuristic algorithms such genetic algorithms (GA), particle swarm optimization (PSO), differential evolution variants L-SHADE and LSHADE-EpSin, whale optimization algorithm (WOA), sine-cosine algorithm (SCA), Harris' hawk optimization (HHO), and equilibrium optimizer (EO). The experimental results suggest that AOA is a high-performance optimization tool with respect to convergence speed and exploration-exploitation balance, as it is effectively applicable for solving complex problems. The source code is currently available for public from: https://www.mathworks.com/matlabcentral/fileexchange/79822-archimedes-optimization-algorithm

Keywords Archimedes' principle · Buoyant force · Optimization · Metaheuristic · Exploration and exploitation

1 Introduction

Over the past few decades, the numerical optimization problems have become increasingly complex and require highly efficient methods to solve. For example, design cost problems in engineering and accuracy problems in data mining often demand methods to find optimum from a large number of available solutions, without wasting efforts in searching sub-optimal regions. Due to complex nature and highly non-convex landscapes, the search-space related to these problems pose several challenges to optimization methods [1]. These include exceptionally complex modalities of the search environment; proportional to the problem size, as well as, growing problem dimensionality [2]. The conventional deterministic methods, based on simple calculus rules perform exhaustive search, cannot produce as efficient solutions as heuristic (trial-and-error) methods within limited computational resources. The non-deterministic methods, also called metaheuristic algorithms, have generated extraordinary results in several practical real-world optimization problems [3, 4]. Numerous applications of these methods can be found in extensive literature related to metaheuristic research. To name a few domains, drug design [5], feature selection [6], motif discovery problem [7, 8], engineering, medical, agriculture, finance and economics are among the beneficiaries [9, 10].

Amongst some of the most successful metaheuristic algorithms are simulated annealing (SA) [11], particle swarm optimization (PSO) [12], genetic algorithms (GA) [13], ant colony optimization (ACO) [14], and artificial bee colony (ABC) [15]. Whereas, some recently introduced methods have also earned adequate attention among researchers due to their efficient problem solving ability;

✉ Essam H. Houssein
essam.halim@mu.edu.eg

[1] Faculty of Engineering, Helwan University, Helwan, Egypt

[2] Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, Sichuan, China

[3] Faculty of Computers and Information, Minia University, Minia, Egypt

[4] Faculty of Engineering, Misr University for Science and Technology, 6th of October, Egypt

Published online: 29 September 2020

such as, Grey wolf optimization (GWO) [16], Harris' hawks optimizer (HHO) [4], bacterial foraging optimization (BFO) [17], moth-flame optimization (MFO) [18], salp swarm algorithm (SSA) [19], and whale optimization algorithm (WOA) [20].

Generally speaking, algorithms that maintain simplicity, ease of implementation as well as ability to avoid local optima are those which stand out among others. This is the reason despite increasingly introduced novel metaheuristic algorithms, only a few retain interest in research community while others often vanish. Among other features, trade-off balance between exploration and exploitation is always critical for metaheuristic algorithms [21]. The algorithms that maintain such balance on a large variety of optimization problems, are the successful ones. Nevertheless, a general procedure of any metaheuristic algorithm can be outlined as Algorithm 1.

Algorithm 1 Pseudo-code of a metaheuristic algorithm.

```
procedure METAHEURISTIC(params)
    initialize population of candidate solutions
    evaluate the initial solutions and remember the best
    one
    while termination criteria not met do
        generate new solutions by modifying existing
        ones
        evaluate new solutions
        if new solutions are better than existing then
            update population
        end if
        remember the best solution found so far
    end while
    return the best solution found
end procedure
```

Besides animals and inspects, powerful phenomena from physics, chemistry, and mathematics have been derived for producing novel optimization methods. The prominent methods devised by the inspirations borrowed from the discipline of Physics are: charged system search (CSS) [22] follows Coulomb and Newtonian's laws, gravitational search algorithm (GSA) [23] based on gravitational theory, ray optimization (RO) [24] observes ray theory and Henry gas solubility optimization (HGSO) based on Henry law [25]. The optimization methods introduced on the basis of Chemistry is chemical reaction optimization (CRO) [26] which mimics molecular interactions in a chemical reaction. Sine-cosine algorithm (SCA) [27] utilizes trigonometry functions and stochastic fractal search (SFS) [28] employs a mathematical concept called fractals.

Essentially, the laws of physics have already produced significant outcomes when formulated into optimization techniques; examples are: thermal exchange optimization (TEO) [29] utilizes Newtonian's laws of cooling, lightening search algorithm (LSA) [30] is inspired by the lightening phenomena in nature, atom search optimization (ASO) [31], equilibrium optimization (EO) [32], magnetic optimization algorithm (MOA) [33] borrows idea from Bict-Savart law of electromagnetism, electromagnetic field optimization (EFO) [34] and ions motion optimization (IMO) [35] are based on attraction-repulsion mechanism of electromagnets and ions versus cations, respectively. In this vein, this study is motivated to continue taking inspirations from the laws of physics. This time, we present Archimedes optimization algorithm (AOA) which is based on the law of physics called Archimedes' principle.

The metaheuristic algorithms mentioned earlier include those that are simple yet effective while others are complicated as well. Some of these algorithms have proven track record of solving numerous optimization problems while others are being effectively modified for improved search performance. Additionally, there is constant influx of new ideas competing with classic methods like SA, PSO, ABC, and ACO while implemented on complex and highly non-linear optimization problems [36]. Due to no-free-lunch theorem [37] which explains the reason why one algorithm or method cannot outperform others on all optimization problems, the room for improvement in existing methods and the opportunities to introduce new methods will always exist. Because, some algorithms will be good on one types of problems and poor on others. However, successful metaheuristic algorithm may be considered as the one which performs well, or at least produces acceptable solutions, on most of the problems. But, keeping in view the wider spectrum of optimization problems, one cannot guarantee that algorithm has been tried and test rigorously. However, we can validate a metaheuristic performance on commonly agreed test suite. Based on the argument, this research also employs the commonly used test environment for experimenting and evaluating performance of the proposed AOA algorithm.

Archimedes' principle explains the law of buoyancy. It states the relationship between an object immersed in a fluid (let's say water) and buoyant force applied on it. Accordingly, an object's buoyancy is subject to an upward force equal to weight of the fluid displaced. If the object's weight is greater than the weight of fluid displaced, the object will sink. Otherwise, it will float when the weight of object and displaced fluid is same. In AOA, the population individuals are the objects immersed in fluid. These objects have density, volume, and acceleration which play important role in buoyance of an object. The idea of AOA is to reach a point where objects are neutrally buoyant; meaning that the net force of fluid is equal to zero. We examined AOA performance by employing an extensive

test-bed comprising of unconstrained benchmark functions and constrained engineering design problems, and found that the proposed approach is efficient with its global search ability.

To sum up, a new population-based algorithm called Archimedes optimization algorithm (AOA) based on the law of physics known as Archimedes' principle is proposed in this paper to compete with the state-of-the-art and recent optimization algorithms, including other physics-inspired methods. It is worth mentioning that the presented algorithm maintains balance between exploration and exploitation. This characteristic makes AOA suitable for solving complex optimization problems with many local optimal solutions because it keeps a population of solutions and investigates a large area to find the best global solution. In summary, the main contributions of this research are as follows:

1. We propose a new population-based algorithm, namely Archimedes optimization algorithm (AOA), which mimics the Archimedes' principle.
2. The statistical significance, convergence ability, exploitation - exploration ratio and the diversity of AOA solutions are evaluated.
3. A series of experiments, comprising of CEC'17 test suite and real-world engineering design problems, are performed to investigate the impact of the proposed algorithm over a challenging test suite in metaheuristic literature.
4. The search efficiency of AOA is validated against well-established algorithms like GA, PSO, L-SHADE, and LSHADE-EpSin, as well as, recent additions like WOA, SCA, HHO, and EO.

This paper is organized as follows. Section 2 explains the basics of Archimedes' principle and design framework of the proposed AOA algorithm. The concept of exploration and exploitation is given in Section 3 which also presents a way to measure these important features of an optimization algorithm. The empirical evaluation of AOA on extensive test environment is given in Section 4. Here, a comprehensive analysis and comparison is also made against the selected metaheuristic algorithms. The final discussion and conclusion is presented in Section 5.

2 Design framework

The proposed Archimedes optimization algorithm (AOA) generally emulates what happens when objects of different weights and volumes are immersed in a fluid. It captures the related phenomenon explained by Archimedes' principle which is described in the following subsection. Next, the implementation of this law of physics in terms of an optimization algorithm is explained.

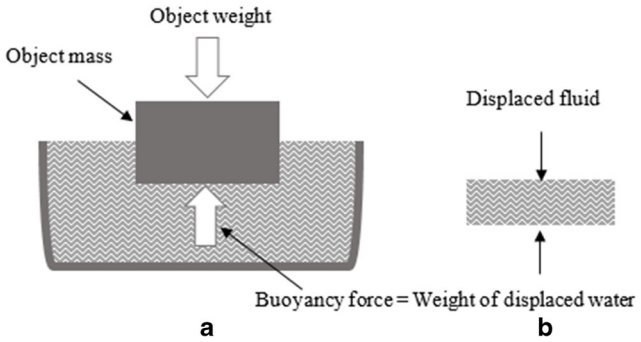


Fig. 1 **a** An object is immersed in a fluid, and **b** The volume of fluid displaced

2.1 Archimedes' principle

Archimedes' principle states that when an object is completely or partially immersed in a fluid, the fluid exerts an upward force on the object equal to weight of the fluid displaced by the object. Figure 1 shows that when an object is immersed in a fluid, it will be experienced by an upward force, called buoyant force, equal to weight of the fluid displaced by the object [38].

2.1.1 Theory

Assume that many objects immersed in the same fluid (Fig. 2) and each one tries to reach the equilibrium state. The immersed objects have different densities and volumes that cause different accelerations.

The object will be in the equilibrium state if the buoyant force F_b is equal to the object's weight W_o:

$$\begin{aligned} F_b &= W_o, \\ p_b v_b a_b &= p_o v_o a_o \end{aligned} \tag{1}$$

where p is the density, v is the volume, and a is the gravity or acceleration, subscripts b and o are for fluid and immersed object, respectively. This equation can be rearranged as:

$$a_o = \frac{p_b v_b a_b}{p_o v_o} \tag{2}$$

Fig. 2 Many objects immersed in the same fluid

If there is another force influenced on the object like collision with another neighbouring object (r), the equilibrium state will be:

$$\begin{aligned} F_b &= W_o, \\ W_b - W_r &= W_o, \\ p_b v_b a_b - p_r v_r a_r &= p_o v_o a_o \end{aligned} \tag{3}$$

2.2 Archimedes optimization algorithm (AOA)

AOA is a population-based algorithm. In the proposed approach, the population individuals are the immersed objects. Like other population-based metaheuristic algorithms, AOA also commences search process with initial population of objects (candidate solutions) with random volumes, densities, and accelerations. At this stage, each object is also initialized with its random position in fluid. After evaluating the fitness of initial population, AOA works in iterations until termination condition meets. In every iteration, AOA updates the density and volume of every object. The acceleration of object is updated based on condition of its collision with any other neighbouring object. The updated density, volume, acceleration determines the new position of an object. Following is the detailed mathematical expression of AOA steps.

2.2.1 Algorithmic steps

In this section, we introduce mathematical formulation of the AOA algorithm. Theoretically, AOA can be considered as a global optimization algorithm as it encompasses both exploration and exploitation processes. Algorithm 2 presents the pseudo-code of the proposed algorithm; including population initialization, population evaluation, and updating parameters. Mathematically, steps of the proposed AOA are detailed as following.

Step 1—Initialization Initialize the positions of all objects using (4):

$$O_i = lb_i + rand \times (ub_i - lb_i); i = 1, 2, ..., N \tag{4}$$

where O_i is the ith object in a population of N objects. lb_i and ub_i are the lower and upper bounds of the search-space, respectively.

Initialize volume (vol) and density (den) for each ith object using (5):

$$\begin{aligned} den_i &= rand \\ vol_i &= rand \end{aligned} \tag{5}$$

where $rand$ is a D dimensional vector randomly generates number between [0, 1]. And finally, initialize acceleration (acc) of ith object using (6):

$$acc_i = lb_i + rand \times (ub_i - lb_i) \tag{6}$$

In this step, evaluate initial population and select the object with the best fitness value. Assign x_{best}, den_{best}, vol_{best}, and acc_{best}.

Step 2—Update densities, volumes The density and volume of object i for the iteration $t + 1$ is updated using (7):

$$\begin{aligned} den_i^{t+1} &= den_i^t + rand \times (den_{best} - den_i^t) \\ vol_i^{t+1} &= vol_i^t + rand \times (vol_{best} - vol_i^t) \end{aligned} \tag{7}$$

where vol_{best} and den_{best} are the volume and density associated with the best object found so far, and $rand$ is uniformly distributed random number.

Step 3—Transfer operator and density factor In the beginning, collision between objects occurs and, after a period of time, the objects try to reach at equilibrium state. This is implemented in AOA with the help of transfer operator TF which transforms search from exploration to exploitation, defined using (8):

$$TF = \exp\left(\frac{t - t_{\max}}{t_{\max}}\right) \tag{8}$$

where transfer TF increases gradually with time until reaching 1. Here t and $t_{\max}$ are iteration number and maximum iterations, respectively. Similarly, density decreasing factor d also assists AOA on global to local search. It decreases with time using (9):

$$d^{t+1} = \exp\left(\frac{t_{\max} - t}{t_{\max}}\right) - \left(\frac{t}{t_{\max}}\right) \tag{9}$$

where d^{t+1} decreases with time that gives the ability to converge in already identified promising region. Note that proper handling of this variable will ensure balance between exploration and exploitation in AOA.

Step 4.1—Exploration phase (collision between objects occurs) If $TF \leq 0.5$, collision between objects occurs, select a random material (mr) and update object's acceleration for iteration $t + 1$ using (10):

$${acc_i}^{t+1} = \frac{den_{mr} + vol_{mr} \times acc_{mr}}{den_i^{t+1} \times vol_i^{t+1}} \tag{10}$$

where den_i, vol_i, and acc_i are density, volume, and acceleration of object i. Whereas acc_{mr}, den_{mr} and vol_{mr} are the acceleration, density, and volume of random material. It is important to mention that $TF \leq 0.5$ ensures exploration during one third of iterations. Applying value other than 0.5 will change exploration-exploitation behavior.

Step 4.2—Exploitation phase (no collision between objects) If $TF > 0.5$, there is no collision between objects, update object's acceleration for iteration $t+1$ using (11):

$$acc_i^{t+1} = \frac{den_{best} + vol_{best} \times acc_{best}}{den_i^{t+1} \times vol_i^{t+1}} \tag{11}$$

where acc_{best} is the acceleration of the best object.

Step 4.3—Normalize acceleration Normalize acceleration to calculate the percentage of change using (12):

$$acc_{i-norm}^{t+1} = u \times \frac{acc_i^{t+1} - \min(acc)}{\max(acc) - \min(acc)} + l \tag{12}$$

where u and l are the range of normalization and set to 0.9 and 0.1, respectively. The $acc_{i,norm}^{t+1}$ determines the percentage of step that each agent will change. If the object i is far away from global optimum, acceleration value will be high—meaning that the object will be in the exploration phase; otherwise, in exploitation phase. This illustrates how the search transforms from exploration to exploitation phase. In normal case, the acceleration factor begins with large value and decreases with time. This helps search agents move towards the global best solution and at the same time they move away from local solutions. But, it is noteworthy to mention that there may remain a few search agents that need more time to stay in exploration phase than normal case. Hence, AOA achieves the balance between exploration and exploitation.

Step 5—Update position If $TF \leq 0.5$ (exploration phase), the i^{th} object's position for next iteration $t+1$ using (13)

$$x_i^{t+1} = x_i^t + C_1 \times rand \times acc_{i-norm}^{t+1} \times d \times (x_{rand} - x_i^t) \tag{13}$$

where C_1 is constant equals to 2. Otherwise, if $TF > 0.5$ (exploitation phase), the objects update their positions using (14).

$$x_i^{t+1} = x_{best}^t + F \times C_2 \times rand \times acc_{i-norm}^{t+1} \times d \times (T \times x_{best} - x_i^t) \tag{14}$$

where C_2 is a constant equals to 6. T increases with time and it is directly proportional to transfer operator and it is defined using $T = C_3 \times TF$. T increases with time in range $[C_3 \times 0.3,\ 1]$ and takes a certain percentage from the best position, initially. It starts with low percentage as this results in large difference between best position and current position, consequently step-size of random walk will be high. As the search proceeds, this percentage increases gradually to decrease difference between the best position and the current position. This leads to achieving an appropriate balance between exploration and exploitation.

F is the flag to change the direction of motion using (15):

$$F = \begin{cases} +1 \text{ if } P \leq 0.5 \\ -1 \text{ if } P > 0.5 \end{cases} \tag{15}$$

where $P = 2 \times rand - C_4$.

Step 6—Evaluation Evaluate each object using objective function f and remember the best solution found so far. Assign x_{best}, den_{best}, vol_{best}, and acc_{best}.

Algorithm 2 Pseudo code of AOA.

procedure AOA(population size N, maximum iterations $t_{\max}$, C_1, C_2, C_3, and C_4)
  Initialize objects population with random positions, densities and volumes using (4), (5), and (6), respectively.
  Evaluate initial population and select the one with the best fitness value.
  Set iteration counter $t = 1$
  while $t \leq t_{max}$ **do**
    for each object i **do**
      Update density and volume of each object using (7)
      Update transfer and density decreasing factors TF and d using (8) and (9), respectively.
      if $TF \leq 0.5$ **then** ▷ Exploration phase
        Update acceleration using (10) and normalize acceleration using (12)
        Update position using (13)
      else ▷ Exploitation phase
        Update acceleration using (11) and normalize acceleration using (12)
        Update direction flag F using (15)
        Update position using (14)
      end if
    end for
    Evaluate each object and select the one with the best fitness value.
    Set $t = t + 1$
  end while
  return object with best fitness value.
end procedure

2.2.2 Sensitivity analysis

Generally, full factorial and fractional factorial design techniques are applied for parameter sensitivity analysis in an algorithm, however high computational cost becomes a major limitation in this regard. Secondly, because metaheuristic algorithms are stochastic in nature, and generate varying solutions each time executed, running them multiple times when considering full factorial designs for evaluating all parameter combinations for all test functions will be nearly infeasible for the length of an initial study [39], like in this case. Therefore, in this subsection, we provide general parameter configuration guidance for AOA control variables C_1 to C_4. We perform sensitivity analysis, with limited full factorial design approach, using three functions selected from three different categories of CEC'17

Table 1 Sensitivity analysis for AOA parameters under different scenarios

Scenarios	Parameters values				Cost function value (Dim = 50)		
	C_1	C_2	C_3	C_4	f_5	f_{12}	f_{26}
1	1	2	1	0.5	9.08E+02	7.20E+09	1.16E+04
2	1	2	1	1	1.03E+03	2.72E+10	1.34E+04
3	1	2	2	0.5	7.71E+02	4.06E+07	1.29E+04
4	1	2	2	1	8.39E+02	9.22E+07	1.14E+04
5	1	4	1	0.5	8.63E+02	5.93E+09	1.15E+04
6	1	4	1	1	1.06E+03	2.97E+10	1.38E+04
7	1	4	2	0.5	8.21E+02	3.70E+07	1.06E+04
8	1	4	2	1	8.25E+02	2.19E+07	1.10E+04
9	1	6	1	0.5	8.91E+02	3.59E+09	8.04E+03
10	1	6	1	1	1.00E+03	1.22E+10	1.42E+04
11	1	6	2	0.5	8.22E+02	2.68E+07	1.09E+04
12	1	6	2	1	8.16E+02	1.04E+08	1.14E+04
13	2	2	1	0.5	8.00E+02	2.13E+08	1.02E+04
14	2	2	1	1	9.96E+02	4.22E+09	1.21E+04
15	2	2	2	0.5	7.66E+02	3.24E+08	8.73E+03
16	2	2	2	1	8.07E+02	4.94E+09	1.13E+04
17	2	4	1	0.5	7.85E+02	6.15E+09	1.09E+04
18	2	4	1	1	9.64E+02	4.54E+09	1.22E+04
19	2	4	2	0.5	7.73E+02	1.80E+07	1.01E+04
20	2	4	2	1	7.79E+02	3.07E+07	1.06E+04
21	2	6	1	0.5	8.00E+02	5.55E+08	1.17E+04
22	2	6	1	1	9.51E+02	7.96E+09	9.98E+03
23	**2**	**6**	**2**	**0.5**	**7.56E+02**	**8.62E+06**	**7.13E+03**
24	2	6	2	1	7.85E+02	7.33E+07	9.92E+03

Bold entries highlight the best results achieved by a particular algorithm on a particular problem

test suite. The selected functions are 50 dimensional Shifted and Rotated Rastrigin's function (f_5), Hybrid Function 2 (N = 3) (f_{12}), and Composite Function 6 (N = 5) (f_{26}). The range of values for these parameters are as followings: $C_1 \in \{1, 2\}$, $C_2 \in \{2, 4, 6\}$, $C_3 \in \{1, 2\}$, and $C_4 \in \{0.5, 1\}$; however, based on optimization problem landscape and difficulty, different values may also be experimented. In our preliminary testing, based on several scenarios given in Table 1 and illustrated via Fig. 3, it is clear that scenario number 23 is the best values for these parameters. With parameter settings as $C_1 = 2$, $C_2 = 6$, $C_3 = 2$, and $C_4 = 0.5$, AOA achieved the best cost function values.

3 Exploration and exploitation

The efficient search ability of a metaheuristic algorithm heavily relies on two essential foundations: exploration and

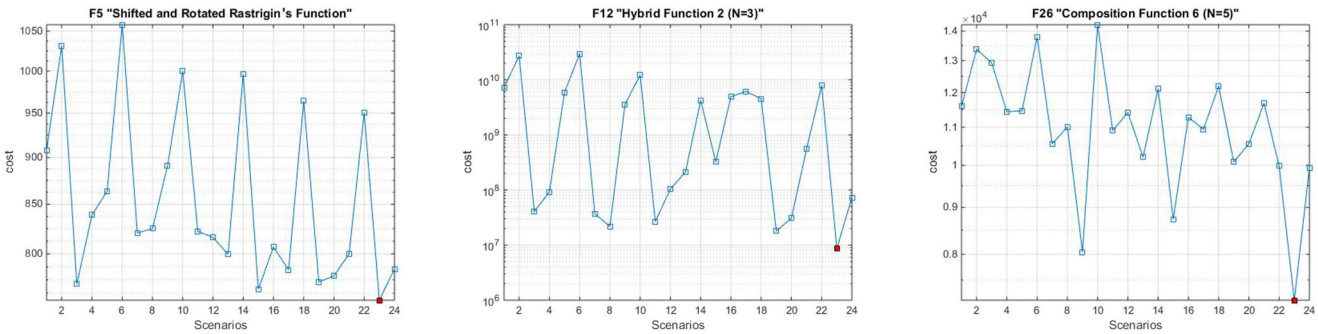


Fig. 3 Cost function values achieved for different scenarios presented in Table 1 pertaining to AOA parameters C_1 to C_4

Table 2 Parameter settings of AOA and selected algorithms

Algorithms	Parameters
GA	Population-size = 30
	Crossover = Whole arithmetic, Probability = 0.8
	Mutation = Gaussian, Probability = 0.05
PSO	Swarm size = 30
	Inertia weight decreases linearly from 0.9 to 0.4 (Default)
	C_1 (individual-best acceleration factor) increases linearly from 0.5 to 2.5 (Default)
	C_2 (global-best acceleration factor) decreases linearly from 2.5 to 0.5 (Default)
L-SHADE	Population-size = 30
	Pbest = 0.1, Arc rate = 2
LSHADE-EpSin	Population-size = 30
	Pbest = 0.1, Arc rate = 2
WOA	Whales Number = 30
	a variable decreases linearly from 2 to 0
	$a2$ linearly decreases from -1 to -2
SCA	Number of agents = 30
	Number of elites = 2
HHO	Harris Hawk Number = 30
	E_0 variable changes from -1 to 1 (Default)
EO	Number of particles = 30
	Generation probability (GP) = 0.5
	$\alpha_1 = 2, \alpha_2 = 1$
AOA	Objects Number = 30
	$C_1 = 2, C_2 = 6$
	$C_3 = 2$ and $C_4 = 0.5$ (CEC and engineering problems)

exploitation [21, 40, 41]. Exploration refers to the search in far reached neighbourhoods for finding global optimum, whereas exploitation is focusing search on already identified potential neighbourhood for converging to optimum solution. Generally, a metaheuristic algorithm starts search process with more exploration and less exploitation; but gradually, the ratio inverts as the search progresses towards the end [44]. Accordingly, the population individuals need to spread all over the search-space and gradually converge to the promising region.

It is crucial to maintain trade-off balance between the two contradictory abilities. This is possible by measuring exploration and exploitation, to adjust search mechanism. The empirical analysis of population diversity, using exploration and exploitation measurements, can be performed when solving a variety of optimization problems. To determine local search ability, problems with unimodal nature are suitable. While, for analysing global search ability, multimodal optimization problems are considered. This study obtained exploration and exploitation measurement while solving a variety of optimization problems, using the approach presented by K. Hussain et al. [41]. The research used dimension-wise diversity measurement. According to the method, the increased average distance within a dimension means exploration, whereas the decreased distance refers to exploitation which suggests that the population individuals are close to each other on the search space. In case of insignificant average diversity within certain iterations, it is suggested that the population has converged. Equation (16) explains the process of measuring dimension-wise diversity:

$$\begin{aligned} Div_j &= \frac{1}{N}\sum_{i=1}^{N} \text{median}(x^j) - x_i^j; \\ Div^t &= \frac{1}{Dim}\sum_{j=1}^{Dim} Div_j \end{aligned} \tag{16}$$

In (16), x_i^j denotes the jth dimension of the ith population individual and median(x^j) the median value of the jth dimension of whole population with N individuals. Div_j denotes the average diversity for dimension j and Div^t the average population diversity for iteration t. When the diversity is measured for all iterations, the percentage of exploration and exploitation can be achieved using (17):

$$\begin{aligned} Exploration\% &= \frac{Div^t}{Div_{max}} \times 100; \\ Exploitation\% &= \frac{|Div^t - Div_{max}|}{Div_{max}} \\ &\quad \times 100, t = 1, 2, ..., t_{max} \end{aligned} \tag{17}$$

Table 3 Statistical results obtained for the CEC'17 functions with Dim = 30

Fun.	Measure	GA	PSO	L-SHADE	LSHADE-EpSin	WOA	SCA	HHO	EO	AOA
f_1	Mean	4.69E+10	3.57E+03	2.74E+09	5.34E+10	1.08E+09	3.60E+03	3.17E+07	6.40E+03	**2.71E+03**
	Std. Dev.	7.50E+09	3.52E+03	1.41E+09	4.17E+09	3.15E+08	2.31E+09	7.47E+06	6.54E+03	1.25E+04
f_3	Mean	1.71E+08	1.45E+04	1.21E+05	1.23E+05	2.95E+05	6.66E+04	3.99E+04	2.69E+04	**1.27E+04**
	Std. Dev.	2.13E+08	7.15E+03	3.94E+04	1.52E+04	7.46E+04	1.33E+04	6.79E+03	8.38E+03	1.29E+04
f_4	Mean	1.88E+04	4.94E+02	8.18E+02	1.18E+04	8.74E+02	2.35E+03	5.55E+02	4.99E+02	**4.79E+02**
	Std. Dev.	3.59E+03	2.55E+01	1.56E+02	2.22E+03	4.64E+01	6.92E+02	1.46E+01	2.48E+01	2.90E+00
f_5	Mean	8.41E+02	6.32E+02	7.65E+02	9.45E+02	8.58E+02	8.14E+02	7.40E+02	6.77E+02	**6.22E+02**
	Std. Dev.	3.36E+01	3.51E+01	1.92E+01	1.75E+01	5.22E+01	2.64E+01	4.90E+01	2.44E+01	2.45E+01
f_6	Mean	6.67E+02	6.41E+02	6.33E+02	6.91E+02	6.91E+02	6.60E+02	6.73E+02	**6.01E+02**	6.18E+02
	Std. Dev.	5.77E+00	1.08E+01	9.51E+00	5.61E+00	1.52E+01	6.28E+00	9.36E+00	8.31E-01	7.04E+00
f_7	Mean	1.08E+03	9.27E+02	1.08E+03	1.19E+03	1.29E+03	1.21E+03	1.30E+03	8.98E+02	**8.66E+02**
	Std. Dev.	3.33E+01	1.18E+01	5.43E+01	1.43E+01	5.18E+01	4.61E+01	5.43E+01	3.33E+01	4.45E+01
f_8	Mean	1.09E+03	9.86E+02	1.05E+03	1.20E+03	1.07E+03	1.09E+03	1.05E+03	8.90E+02	**8.85E+02**
	Std. Dev.	2.69E+01	1.14E+01	2.45E+01	2.60E+01	4.81E+01	2.13E+01	3.46E+01	1.81E+01	1.54E+01
f_9	Mean	1.24E+04	3.23E+03	5.66E+03	1.51E+04	1.16E+04	7.92E+03	7.55E+03	2.40E+03	**1.26E+03**
	Std. Dev.	2.11E+03	1.25E+03	2.76E+03	1.32E+03	5.87E+03	1.54E+03	1.90E+03	4.48E+02	5.72E+02
f_{10}	Mean	8.06E+03	4.80E+03	7.69E+03	8.59E+03	6.73E+03	8.66E+03	6.14E+03	5.21E+03	**5.10E+03**
	Std. Dev.	3.16E+02	6.05E+02	2.72E+02	2.31E+02	9.07E+02	3.33E+02	3.57E+02	7.53E+02	5.98E+02
f_{11}	Mean	4.66E+04	**1.24E+03**	2.52E+03	7.33E+03	6.90E+03	3.45E+03	1.29E+03	1.29E+03	**1.24E+03**
	Std. Dev.	3.71E+04	4.34E+01	1.01E+03	2.13E+03	3.18E+03	1.08E+03	3.41E+01	4.29E+01	4.68E+01
f_{12}	Mean	1.10E+10	**1.17E+05**	9.73E+07	6.91E+09	3.17E+08	2.15E+09	3.69E+07	7.58E+05	3.53E+05
	Std. Dev.	1.14E+09	1.04E+05	2.44E+07	1.20E+09	1.29E+08	5.75E+08	2.63E+07	7.77E+05	6.59E+05
f_{13}	Mean	2.34E+10	1.56E+04	3.23E+06	3.41E+09	1.21E+06	9.15E+08	6.68E+05	1.79E+04	**5.13E+03**
	Std. Dev.	4.07E+09	1.64E+04	3.46E+06	9.54E+08	3.84E+05	4.21E+08	4.26E+05	1.68E+04	3.62E+03
f_{14}	Mean	6.95E+07	3.06E+04	4.66E+05	7.34E+06	9.82E+06	1.12E+06	9.12E+06	1.19E+05	**4.24E+04**
	Std. Dev.	3.95E+07	3.04E+04	3.46E+05	4.89E+05	2.59E+06	3.85E+05	2.54E+05	3.70E+04	1.25E+04
f_{15}	Mean	3.16E+09	1.17E+04	3.10E+05	3.49E+08	2.27E+06	4.26E+07	1.11E+05	4.79E+03	**2.74E+03**
	Std. Dev.	1.59E+09	1.07E+04	3.21E+05	1.31E+08	3.67E+06	3.22E+07	1.13E+05	2.74E+03	1.20E+03
f_{16}	Mean	8.45E+03	2.70E+03	3.96E+03	4.53E+03	3.74E+03	3.95E+03	3.82E+03	2.42E+03	**2.06E+03**
	Std. Dev.	2.79E+03	2.90E+02	1.90E+02	1.45E+02	1.09E+03	2.38E+02	2.66E+02	3.42E+02	2.66E+02
f_{17}	Mean	3.02E+04	2.23E+03	2.65E+03	3.28E+03	2.91E+03	2.64E+03	2.50E+03	2.12E+03	**2.09E+03**
	Std. Dev.	4.81E+04	1.93E+02	1.40E+02	1.95E+02	3.23E+02	2.88E+02	3.09E+02	2.19E+02	1.43E+02
f_{18}	Mean	4.68E+08	**1.90E+05**	3.97E+05	1.94E+07	1.24E+07	9.49E+06	2.54E+06	7.54E+05	**1.90E+05**
	Std. Dev.	2.64E+08	2.30E+05	1.81E+05	1.16E+07	6.47E+06	4.99E+06	1.02E+06	5.47E+05	2.68E+05
f_{19}	Mean	3.19E+09	4.91E+08	3.94E+06	6.36E+06	9.32E+06	8.47E+07	5.64E+05	3.03E+04	**5.65E+03**
	Std. Dev.	1.77E+09	1.32E+04	4.13E+05	9.88E+07	5.06E+06	4.12E+07	5.59E+05	2.03E+03	2.12E+03
f_{20}	Mean	3.09E+03	2.53E+03	3.01E+03	2.91E+03	2.82E+03	2.90E+03	2.85E+03	**2.29E+03**	2.48E+03
	Std. Dev.	1.63E+02	1.81E+02	1.98E+02	1.10E+02	1.98E+02	1.43E+02	1.86E+02	1.26E+02	1.92E+02
f_{21}	Mean	2.64E+03	2.44E+03	2.54E+03	2.72E+03	2.60E+03	2.59E+03	2.61E+03	2.44E+03	**2.35E+03**
	Std. Dev.	4.64E+01	5.07E+01	1.45E+01	1.36E+01	2.00E+01	2.62E+01	4.52E+01	2.02E+01	2.16E+01
f_{22}	Mean	8.41E+03	4.80E+03	4.79E+03	8.26E+03	6.55E+03	9.01E+03	5.77E+03	**3.51E+03**	7.16E+03
	Std. Dev.	2.18E+03	2.30E+03	2.98E+03	5.41E+02	2.45E+03	2.49E+03	2.99E+03	1.97E+03	2.35E+03
f_{23}	Mean	3.50E+03	2.87E+03	2.93E+03	3.30E+03	3.07E+03	3.05E+03	3.25E+03	**2.73E+03**	2.87E+03
	Std. Dev.	1.85E+02	7.21E+01	2.35E+01	2.98E+01	1.06E+02	2.96E+01	6.49E+01	1.92E+01	7.22E+01
f_{24}	Mean	3.51E+03	3.00E+03	3.09E+03	3.53E+03	3.23E+03	3.24E+03	3.43E+03	2.99E+03	**2.97E+03**
	Std. Dev.	1.20E+02	6.30E+01	2.72E+01	4.78E+01	6.98E+01	3.01E+01	1.17E+02	2.34E+01	3.19E+01
f_{25}	Mean	8.40E+03	**2.91E+03**	3.18E+03	6.26E+03	3.10E+03	3.48E+03	2.94E+03	2.95E+03	**2.91E+03**
	Std. Dev.	1.78E+03	2.08E+01	1.30E+02	4.10E+02	3.28E+01	1.43E+02	1.62E+01	2.73E+00	1.78E+01

Table 3 (continued)

Fun.	Measure	GA	PSO	L-SHADE	LSHADE-EpSin	WOA	SCA	HHO	EO	AOA
f_{26}	Mean	1.34E+04	5.87E+03	6.41E+03	1.01E+04	9.31E+03	7.60E+03	8.21E+03	4.34E+03	**3.24E+03**
	Std. Dev.	2.38E+03	1.51E+03	5.93E+02	5.89E+02	1.01E+03	3.93E+02	1.18E+03	7.24E+02	1.68E+03
f_{27}	Mean	4.55E+03	3.27E+03	3.32E+03	3.94E+03	3.48E+03	3.51E+03	3.65E+03	3.22E+03	**3.20E+03**
	Std. Dev.	3.38E+02	3.52E+01	3.60E+01	9.09E+01	8.82E+01	4.41E+01	3.28E+02	9.77E+00	2.83E-04
f_{28}	Mean	7.49E+03	**3.22E+03**	3.71E+03	6.50E+03	3.56E+03	4.23E+03	3.35E+03	3.22E+03	3.30E+03
	Std. Dev.	5.16E+02	1.99E+01	1.78E+02	3.88E+02	1.50E+02	2.36E+02	3.48E+01	1.93E+01	1.17E+01
f_{29}	Mean	6.98E+04	3.92E+03	4.64E+03	5.81E+03	5.42E+03	5.05E+03	4.90E+03	**3.68E+03**	**3.68E+03**
	Std. Dev.	5.34E+04	2.22E+02	8.41E+01	8.42E+01	3.70E+02	1.93E+02	6.31E+02	2.27E+02	2.13E+02
f_{30}	Mean	2.75E+09	9.95E+03	7.82E+06	4.12E+08	3.98E+07	1.41E+08	4.63E+06	1.01E+04	**4.05E+03**
	Std. Dev.	6.85E+08	3.85E+03	3.68E+06	1.03E+08	3.06E+07	4.86E+07	3.85E+06	3.45E+03	1.55E+03

Bold entries highlight the best results achieved by a particular algorithm on a particular problem

where Div^t is population diversity of tth iteration and Div_{max} is the maximum diversity found in all t_{max} iterations.

Having discussed the exploration-exploitation measurement using population diversity, it is important to mention that the population diversity does not guarantee that an efficient exploration is being performed by the search agents. However, it definitely clarifies that the search is being performed on distant locations in search space. On the other hand, it also suggests that the candidate solutions under consideration are diversified and not stagnant. Contrarily, population with reduced diversity indicates exploitation; yet, it is essential that the search is being performed around global optimum region. That said, it can be contemplated that exploration-exploitation measurement using population diversity has certain limitations. Nevertheless, in the absence of direct exploration-exploitation measurement approach [42], this study follows the method proposed in [41]. Because, in previous research, mostly population diversity is adopted as a measure for controlling explorative and exploitative search strategies. To the best of authors knowledge, there exists only one research [43] that proposed a direct measure of exploration-exploitation for evolutionary algorithms, calling it ancestry tree-based approach; however, its limitation to evolutionary algorithms motivates this study to apply diversity-based exploration-exploitation measurement on the proposed method.

4 Experimental results

To investigate effectiveness of the proposed AOA algorithm, we have employed 29 test functions of CEC'17 test suite and four constrained engineering design problems, which are widely used in existing empirical literature [16, 20, 23, 45]. All of these functions are minimization problems which are useful for evaluating optimization method characteristics such as search efficiency and convergence rate.

4.1 Parameter settings

Since a metaheuristic algorithm is stochastic in nature, its results may vary each time an algorithm is executed. Therefore, we performed each experiment 30 times with parameter settings mentioned in Table 2. As mentioned in the table, apart from AOA, several other algorithms including well established methods like GA, PSO, L-SHADE, and LSHADE-EpSin, as well as, newly introduced methods like WOA, SCA, HHO, and EO were also used to solve the same experimental suite for comparison purpose. For fair comparison, all the algorithms were run with a maximum of 1000 iterations (30000 function evaluations). The algorithms were programmed in MATLAB 8.0.604 (2014b) 64-bit version and executed on computation environment of Intel Core i7 CPU 2.00GHz, 2.5GHz, 8GB RAM and 64-bit operating system. Besides the parameters of the selected algorithms presented in Table 2, the common settings include population size ($N = 30$), maximum iterations ($t_{\max} = 1000$), and 30 independent runs for each optimization problem.

4.2 CEC 2017 test suite analysis

This study employed more complex optimization problems encompassed in CEC'17 test suite [46] for better performance evaluation of the AOA algorithm. The test-bed comprises of 30 functions in which function f_2 is excluded, hence 29 functions of different modalities and complexities are to be employed for the testing of AOA and other counterpart algorithms selected in this study. Functions from f_1 and f_3 are unimodal, f_4 to f_{10} are multimodal, f_{11} to f_{20} are hybrid, and f_{21} to f_{30} are composition functions. All of the test functions maintain a hyperspace of $[-100,100]^D$. Overall, the CEC'17 test suite is reasonably complex and dynamic, which can be used for extensive study of exploration and exploitation capabilities of a metaheuristic algorithm.

Table 4 Statistical results obtained for the CEC'17 functions with Dim=50

Fun.	Measure	GA	PSO	L-SHADE	LSHADE-EpSin	WOA	SCA	HHO	EO	AOA
f_1	Mean	4.52E+10	1.34E+03	9.14E+09	8.60E+10	8.47E+08	2.57E+10	6.63E+07	6.03E+03	**2.50E+02**
	Std. Dev.	8.37E+09	1.38E+06	1.12E+09	2.26E+09	2.31E+09	8.40E+09	1.21E+08	1.46E+05	7.74E+08
f_3	Mean	3.55E+08	**1.12E+05**	1.30E+05	1.24E+05	2.34E+05	1.75E+05	1.34E+05	1.15E+05	2.01E+05
	Std. Dev.	7.05E+08	3.48E+04	4.88E+04	1.11E+04	3.73E+04	2.04E+04	1.39E+04	2.57E+04	5.32E+04
f_4	Mean	1.83E+04	5.10E+02	7.56E+02	1.04E+04	2.39E+03	1.10E+04	8.22E+02	5.50E+02	**4.80E+02**
	Std. Dev.	3.06E+03	4.47E+01	1.11E+02	1.58E+03	4.12E+02	1.71E+03	6.33E+01	5.14E+01	7.57E+01
f_5	Mean	8.37E+02	7.61E+02	6.91E+02	9.05E+02	9.00E+02	8.02E+02	7.50E+02	6.18E+02	**6.00E+02**
	Std. Dev.	2.72E+01	4.00E+01	2.62E+01	1.25E+01	7.95E+01	3.43E+01	1.75E+01	4.00E+01	3.04E+01
f_6	Mean	6.61E+02	6.45E+02	7.42E+02	6.91E+02	6.91E+02	6.84E+02	6.76E+02	**6.05E+02**	6.37E+02
	Std. Dev.	7.59E+00	8.36E+00	2.62E+01	3.36E+00	1.09E+01	6.61E+00	5.16E+00	4.01E+00	6.52E+00
f_7	Mean	1.35E+03	1.14E+03	**7.42E+02**	2.00E+03	1.86E+03	1.77E+03	1.90E+03	1.05E+03	1.14E+03
	Std. Dev.	3.11E+01	5.81E+01	2.62E+01	4.65E+01	9.36E+01	6.97E+01	4.95E+01	1.03E+02	9.39E+01
f_8	Mean	1.05E+03	9.23E+02	1.02E+03	1.20E+03	1.07E+03	1.07E+03	1.03E+3	9.00E+02	**8.51E+02**
	Std. Dev.	1.91E+01	4.83E+01	2.84E+01	1.17E+01	4.45E+01	2.92E+01	4.63E+01	4.94E+01	3.21E+01
f_9	Mean	1.21E+04	1.19E+04	**1.49E+03**	1.64E+04	2.89E+04	2.99E+04	2.74E+04	3.72E+04	9.26E+03
	Std. Dev.	4.55E+03	6.31E+03	1.10E+03	1.08E+03	8.74E+03	3.85E+03	1.87E+03	1.06E+03	4.94E+03
f_{10}	Mean	8.34E+03	4.61E+03	6.53E+03	8.95E+03	7.47E+03	8.48E+03	1.05E+04	4.56E+03	**4.06E+03**
	Std. Dev.	6.80E+02	7.93E+02	4.49E+02	1.56E+02	1.36E+03	5.23E+02	4.77E+02	6.28E+02	1.57E+03
f_{11}	Mean	5.69E+04	**1.32E+03**	2.15E+03	7.44E+03	4.92E+03	1.21E+04	1.71E+03	7.89E+03	2.15E+05
	Std. Dev.	2.60E+04	4.88E+01	6.36E+02	1.65E+03	1.89E+03	2.83E+03	1.06E+02	7.89E+03	7.86E+05
f_{12}	Mean	1.09E+10	1.10E+05	8.00E+07	8.23E+09	1.84E+08	1.00E+09	9.13E+06	1.09E+05	**7.42E+04**
	Std. Dev.	1.42E+09	1.43E+06	5.17E+07	1.20E+09	1.70E+09	6.64E+09	1.49E+08	3.44E+06	6.74E+06
f_{13}	Mean	1.76E+10	**9.30E+03**	3.58E+06	3.34E+09	1.86E+08	4.53E+09	1.08E+07	2.27E+04	1.41E+04
	Std. Dev.	8.71E+09	6.85E+03	2.95E+06	1.31E+09	1.37E+08	1.61E+09	1.38E+07	9.09E+04	1.66E+04
f_{14}	Mean	6.34E+07	3.96E+04	2.86E+04	2.53E+06	2.01E+06	7.95E+05	2.00E+05	9.93E+04	**9.03E+03**
	Std. Dev.	2.97E+07	7.49E+04	1.56E+05	1.76E+10	7.02E+06	4.05E+06	7.53E+05	3.75E+05	4.25E+05
f_{15}	Mean	3.06E+09	**6.64E+03**	1.74E+05	3.44E+08	1.43E+07	9.78E+08	1.17E+06	3.91E+04	2.01E+04
	Std. Dev.	2.22E+09	4.28E+03	1.29E+05	8.90E+07	1.99E+07	3.26E+08	3.57E+05	2.79E+04	1.18E+04
f_{16}	Mean	6.00E+3	2.09E+03	3.03E+03	5.00E+03	3.00E+03	4.02E+03	3.03E+03	2.08E+03	**2.05E+03**
	Std. Dev.	1.39E+03	4.60E+02	2.99E+02	1.13E+02	1.20E+03	2.95E+02	3.47E+02	6.32E+02	4.24E+02
f_{17}	Mean	1.34E+04	3.24E+03	**2.46E+03**	3.25E+03	4.34E+03	4.82E+03	4.04E+03	3.12E+03	3.07E+03
	Std. Dev.	1.02E+04	2.69E+02	1.50E+02	2.09E+02	3.17E+02	3.56E+02	1.76E+02	3.55E+02	3.01E+02
f_{18}	Mean	9.86E+08	1.27E+05	2.86E+05	1.63E+07	3.75E+07	6.65E+06	2.98E+06	9.26E+05	**8.85E+04**
	Std. Dev.	1.52E+08	4.31E+05	1.62E+06	1.16E+07	4.94E+07	1.46E+07	3.91E+06	2.04E+06	1.12E+06
f_{19}	Mean	4.42E+09	1.02E+04	1.05E+06	5.14E+08	2.34E+06	5.55E+08	3.12E+05	1.01E+04	**3.07E+03**
	Std. Dev.	1.11E+09	8.45E+03	7.89E+05	1.42E+08	1.62E+07	2.02E+08	1.44E+06	1.26E+04	4.92E+03
f_{20}	Mean	3.16E+03	3.32E+03	**3.09E+03**	2.95E+03	3.79E+03	4.18E+03	3.52E+03	3.32E+03	3.32E+03
	Std. Dev.	2.51E+02	3.88E+02	1.66E+02	6.64E+01	4.42E+02	2.29E+02	2.92E+02	3.16E+02	3.46E+02
f_{21}	Mean	2.64E+03	2.43E+03	2.54E+03	2.70E+03	2.58E+03	2.62E+03	2.61E+03	2.39E+03	**2.35E+03**
	Std. Dev.	3.79E+01	6.14E+01	3.31E+01	1.57E+01	1.84E+02	3.18E+01	4.35E+01	2.30E+01	3.47E+01
f_{22}	Mean	8.30E+03	9.40E+03	4.22E+03	8.27E+03	1.41E+04	1.70E+04	1.10E+04	**1.06E+04**	1.23E+04
	Std. Dev.	1.60E+03	1.17E+03	2.70E+03	7.67E+02	1.03E+03	2.86E+02	3.84E+02	1.10E+03	2.36E+03
f_{23}	Mean	3.35E+03	3.18E+03	**2.89E+03**	3.30E+03	3.81E+03	3.70E+03	3.85E+03	3.57E+03	3.18E+03
	Std. Dev.	6.28E+01	9.85E+01	2.53E+01	3.49E+01	2.19E+02	4.76E+01	1.54E+02	3.49E+01	1.54E+02
f_{24}	Mean	5.14E+03	2.85E+03	3.21E+03	3.52E+03	3.48E+03	3.46E+03	3.37E+03	2.84E+03	**2.75E+03**
	Std. Dev.	2.15E+01	7.94E+01	2.14E+01	3.18E+01	2.19E+02	8.27E+01	1.31E+02	2.54E+01	2.33E+02
f_{25}	Mean	3.54E+03	**3.08E+03**	3.07E+03	3.53E+03	3.91E+03	8.92E+03	3.28E+03	3.11E+03	3.16E+03
	Std. Dev.	1.06E+02	2.60E+01	2.81E+01	6.31E+01	2.55E+02	1.82E+03	4.94E+01	2.65E+01	4.90E+01

Table 4 (continued)

Fun.	Measure	GA	PSO	L-SHADE	LSHADE-EpSin	WOA	SCA	HHO	EO	AOA
f_{26}	Mean	1.85E+04	5.28E+03	5.53E+03	1.03E+04	7.10E+03	6.11E+03	9.03E+03	3.34E+03	**1.98E+03**
	Std. Dev.	1.97E+03	2.60E+03	2.31E+02	2.94E+02	2.84E+03	6.26E+02	2.11E+03	4.69E+02	2.76E+03
f_{27}	Mean	4.04E+03	3.63E+03	3.28E+03	3.90E+03	4.81E+03	4.79E+03	5.45E+03	3.42E+03	**3.20E+03**
	Std. Dev.	2.32E+02	1.22E+02	1.69E+01	4.46E+01	6.14E+02	1.82E+02	6.73E+02	7.31E+01	2.64E-04
f_{28}	Mean	7.52E+03	**3.34E+03**	3.86E+03	6.35E+03	5.09E+03	8.26E+03	3.73E+03	3.50E+03	3.46E+03
	Std. Dev.	1.13E+03	4.19E+01	3.90E+02	2.12E+02	3.15E+02	8.18E+02	1.28E+02	4.42E+01	1.70E+02
f_{29}	Mean	4.74E+04	4.83E+03	4.68E+03	6.01E+03	8.89E+03	8.50E+03	6.23E+03	4.38E+03	**4.29E+03**
	Std. Dev.	3.68E+04	4.24E+02	2.72E+02	1.42E+02	1.42E+03	6.96E+02	7.39E+02	3.95E+02	3.37E+02
f_{30}	Mean	6.86E+09	2.34E+04	1.13E+06	4.26E+08	1.88E+07	2.10E+08	1.70E+07	2.03E+04	**3.42E+03**
	Std. Dev.	1.36E+09	1.89E+05	5.05E+06	3.98E+07	1.33E+08	2.90E+08	2.67E+07	4.29E+05	3.87E+03

Bold entries highlight the best results achieved by a particular algorithm on a particular problem

4.2.1 Statistical results

When employing 29 test functions of CEC'17 test suite, the experimental study proved that AOA showed its efficient search performance for this test-bed. Considering the results of 30 and 50 dimensional CEC'17 functions (Tables 3 and 4), AOA outperformed other selected metaheuristic algorithms for 19 and 16 functions, respectively. According to Table 3, EO achieved better optimum values than AOA for four functions (f_6, f_{20}, f_{22}, and f_{23}), while it performed equally well as AOA for f_{29} with 30 dimensions. Similarly, PSO also performed better than AOA for two 30 dimensional CEC'17 functions (f_{12} and f_{18}), while it generated equally better solution as of AOA for functions f_{11}, f_{18}, and f_{25}. The least performers for these functions were GA, L-SHADE, LSHADE-EpSin, WOA, SCA, and HHO. On the other hand, for 50 dimensional CEC'17 functions (Table 4), AOA produced superior results than others for 16 functions. On the other hand, AOA underperformed than EO for f_6 and f_{22}, L-SHADE for f_7, f_9, f_{17}, f_{20}, and f_{23}, PSO for f_3, f_{11}, f_{13}, f_{15}, f_{25}, and f_{28}. In this context, GA, LSHADE-EpSin, WOA, SCA, and HHO methods remained the least performers.

To statistically validate the results achieved by AOA, we performed nonparametric Wilcoxon Ranksum test, as it serves to produce meaningful comparison between the proposed and alternative methods. The p-values for the test are given in Table 5 where $\triangle$, $\triangledown$, and $\approx$ indicate that AOA is significantly better than the alternative method, it is significantly inferior than the other, or insignificantly different from the competitive method, respectively. According to Table 5, AOA results remained significantly better than GA, LSHADE-EpSin, and SCA for all CEC'17 test functions. On the other hand, for majority of the test suite, AOA generated significantly better solutions than WOA and HHO, except for f_{22} where these methods remained insignificantly different. In case of PSO and EO, AOA generated significantly inferior results for five and two functions, respectively, and AOA remained insignificant for four functions; rest, AOA remained significantly superior method than PSO and EO.

4.2.2 Convergence analysis

The convergence ability of AOA and other eight counterparts is depicted via Figs. 4 and 5 for CEC'17 test suite with 30 and 50 dimensions, respectively. In these figures, the convergence curves for the selected functions are presented, since graphs for all 29 functions enlarge the length of the paper significantly. According to convergence curves for nine selected functions f_5, f_8, f_{12}, f_{14}, f_{16}, f_{19}, f_{21}, f_{26} and f_{30} the proposed AOA algorithm showed faster convergence ability. It is because AOA performed search effectively by maintaining trade-off balance between exploration and exploitation. Specially, in case of f_{18}, f_{21}, f_{26} and f_{30} for 30 and 50 dimensions, AOA not only converged faster than other eight algorithms, it also managed to find comparatively much better optimum solutions.

4.2.3 Exploration-exploitation analysis

The statistical results of AOA on CEC'17 presented in this section prove its efficiency, for further extensive analysis, we also recorded exploration-exploitation ratios during search process. To analyze the relevant search behaviors, Fig. 6 illustrates exploration and exploitation maintained by AOA while solving some of the 50 dimensional CEC'17 problems.

As it can be observed from line graphs presented in Fig. 6 that AOA algorithm started with high exploration and low exploitation, but mostly later transformed into exploitation strategy during most of the iterations. However, on f_{18} and f_{20} with 50 dimensions, AOA performed exploration higher than exploitation during searching for global optimum location (Fig. 6). Similar behavior was noticed on f_7, f_{18}, and f_{20} with 50 dimensions (Fig. 6). When observing the general trend, compared to exploration

Table 5 Wilcoxon ranksum test results for AOA vs. other algorithms (CEC'17 functions with Dim = 50)

Fun.	GA	PSO	L-SHADE	LSHADE-EpSin	WOA	SCA	HHO	EO
f_1	3.02E-11	4.57E-09	3.02E-11	3.02E-11	3.02E-11	3.02E-11	4.08E-11	2.19E-08
	△	△	△	△	△	△	△	△
f_3	4.50E-11	3.08E-08	1.41E-04	1.19E-06	1.52E-03	2.97E-01	7.69E-08	2.61E-10
	△	▽	△	△	△	△	△	△
f_4	3.02E-11	5.83E-03	3.02E-11	3.02E-11	3.02E-11	3.02E-11	4.18E-09	6.28E-06
	△	△	△	△	△	△	△	△
f_5	3.02E-11	2.42E-02	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	5.27E-05
	△	△	△	△	△	△	△	△
f_6	3.02E-11	6.38E-03	1.73E-06	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
	△	△	△	△	△	△	△	▽
f_7	3.02E-11	8.20E-07	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.01E-07
	△	△	▽	△	△	△	△	△
f_8	3.02E-11	6.66E-04	3.02E-11	3.02E-11	3.02E-11	3.02E-11	4.08E-11	5.56E-04
	△	△	△	△	△	△	△	△
f_9	3.02E-11	3.02E-11	1.78E-10	3.02E-11	6.07E-11	6.70E-11	3.69E-11	1.29E-09
	△	△	▽	△	△	△	△	△
f_{10}	4.98E-11	8.88E-06	4.50E-11	3.02E-11	7.77E-09	2.15E-10	6.70E-11	6.16E-04
	△	△	△	△	△	△	△	△
f_{11}	2.30E-06	3.01E-11	7.96E-02	6.73E-05	1.30E-01	3.63E-01	8.73E-10	3.00E-11
	△	▽	△	△	△	△	△	△
f_{12}	3.02E-11	1.41E-09	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	9.88E-06
	△	△	△	△	△	△	△	△
f_{13}	3.02E-11	2.40E-01	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	9.12E-01
	△	≈	△	△	△	△	△	≈
f_{14}	3.02E-11	1.86E-09	3.02E-11	3.02E-11	2.92E-09	3.69E-11	2.20E-07	1.75E-05
	△	△	△	△	△	△	△	△
f_{15}	3.02E-11	6.74E-06	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.71E-01
	△	▽	△	△	△	△	△	△
f_{16}	3.02E-11	8.88E-01	3.02E-11	3.02E-11	3.69E-11	3.02E-11	7.39E-11	6.20E-01
	△	△	△	△	△	△	△	≈
f_{17}	3.02E-11	8.19E-01	4.98E-11	3.02E-11	1.96E-10	3.02E-11	8.48E-09	2.97E-01
	△	≈	▽	△	△	△	△	≈
f_{18}	3.02E-11	5.97E-09	2.02E-08	3.02E-11	3.02E-11	3.02E-11	1.16E-07	4.98E-04
	△	△	△	△	△	△	△	△
f_{19}	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
	△	△	△	△	△	△	△	△
f_{20}	6.70E-11	1.68E-03	2.37E-10	8.15E-11	1.78E-04	2.15E-10	2.51E-02	1.95E-03
	△	▽	△	△	△	△	△	△
f_{21}	3.02E-11	6.38E-03	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	9.07E-03
	△	△	△	△	△	△	△	△
f_{22}	5.61E-05	4.18E-09	1.17E-09	3.81E-07	8.30E-01	3.65E-08	5.01E-01	1.39E-06
	△	△	△	△	≈	△	≈	▽
f_{23}	3.02E-11	3.87E-01	6.05E-07	3.02E-11	4.62E-10	6.72E-10	3.34E-11	8.15E-11
	△	≈	▽	△	△	△	△	△
f_{24}	1.69E-09	4.38E-01	4.57E-09	3.02E-11	5.57E-03	3.02E-11	1.33E-10	3.02E-11
	△	≈	△	△	△	△	△	△
f_{25}	3.02E-11	7.38E-10	3.02E-11	3.02E-11	3.02E-11	3.02E-11	4.44E-07	4.11E-07
	△	▽	△	△	△	△	△	△

Table 5 (continued)

Fun.	GA	PSO	L-SHADE	LSHADE-EpSin	WOA	SCA	HHO	EO
f_{26}	3.02E-11	6.55E-04	2.16E-03	3.02E-11	3.02E-11	3.02E-11	1.29E-09	5.01E-04
	△	△	△	△	△	△	△	△
f_{27}	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
	△	△	△	△	△	△	△	△
f_{28}	3.02E-11	7.06E-01	1.78E-10	3.02E-11	1.60E-07	3.02E-11	5.60E-07	9.47E-01
	△	≈	△	△	△	△	△	≈
f_{29}	3.02E-11	1.95E-03	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.69E-11	2.81E-02
	△	△	△	△	△	△	△	△
f_{30}	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
	△	△	△	△	△	△	△	△

and exploitation behavior of AOA, the algorithm maintained relatively dynamic behavior on CEC'17 functions (Fig. 6).

4.3 Engineering design problems

We further tested AOA for solving four constrained engineering design problems: tension/compression spring design, welded beam design, pressure vessel design, and speed reducer design.

4.3.1 Welded beam design problem

The first problem is to minimize the cost of welded beam design (Fig. 7). Coello [47] first proposed this

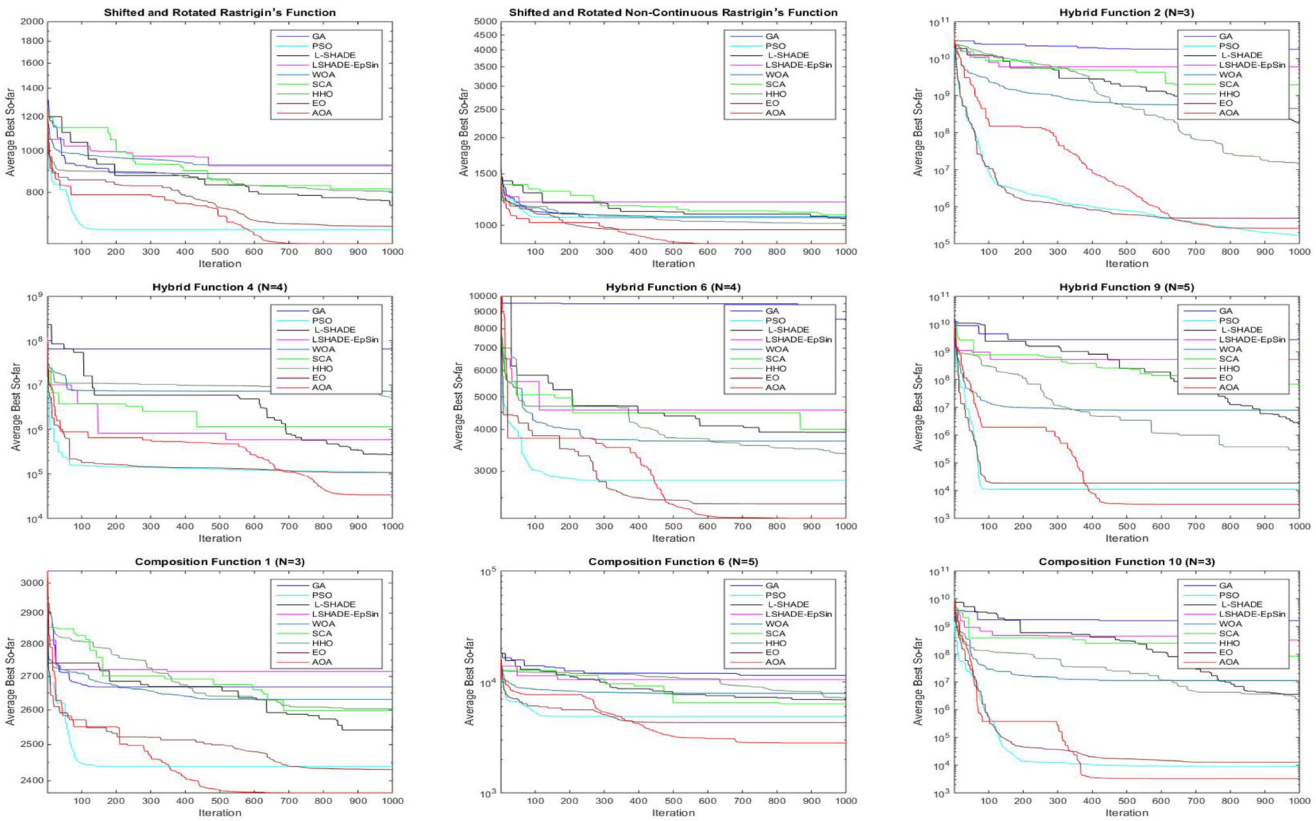


Fig. 4 Convergence curves of competitor algorithms on CEC'17 functions with 30 dimensions

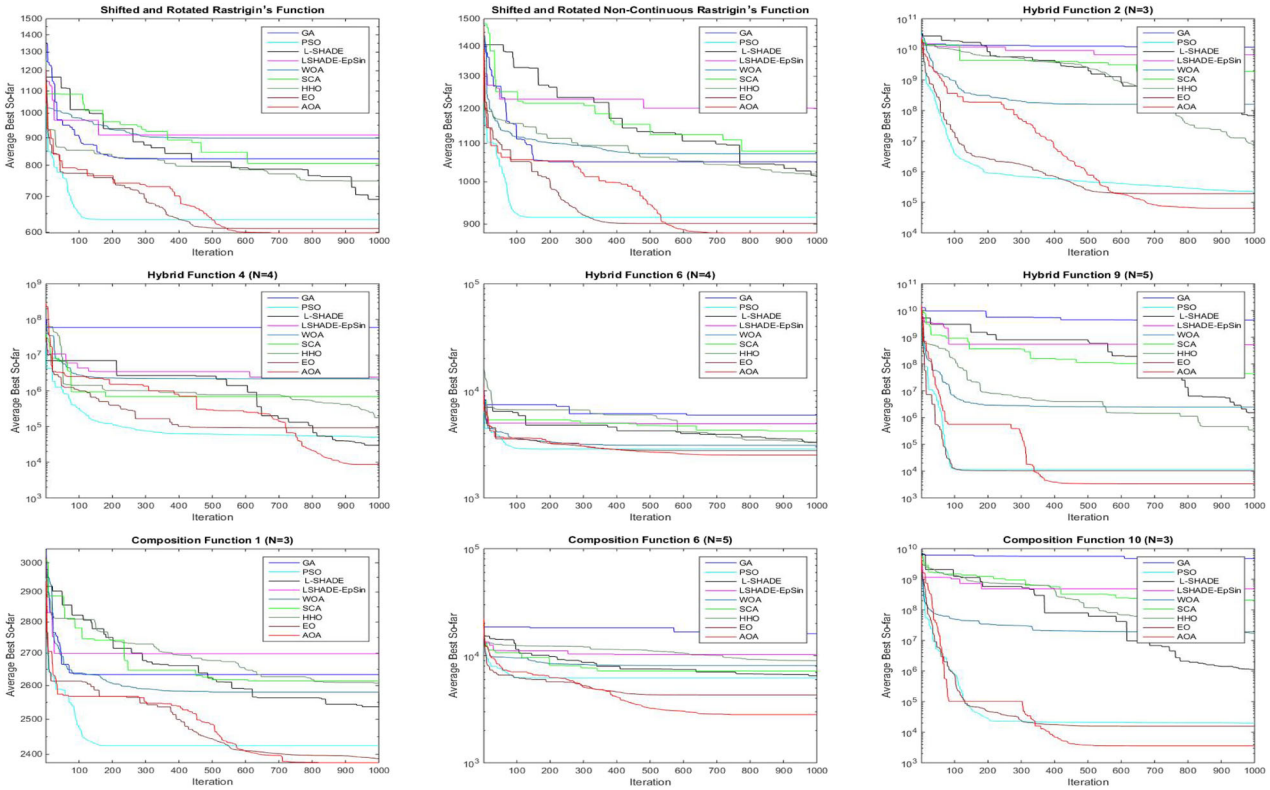


Fig. 5 Convergence curves of competitor algorithms on CEC'17 functions with 50 dimensions

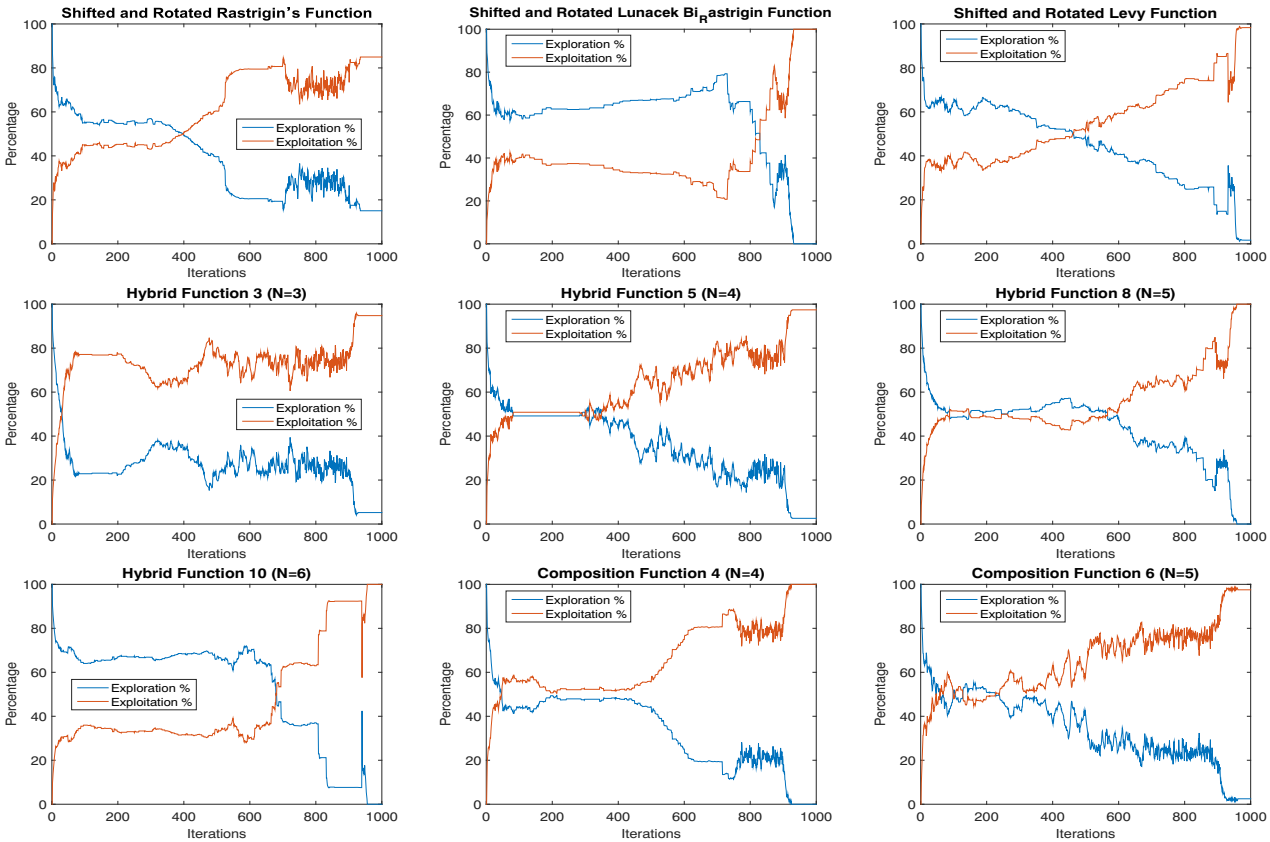
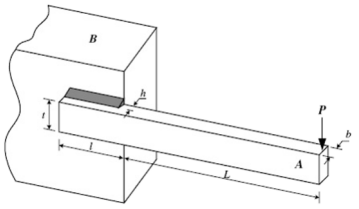
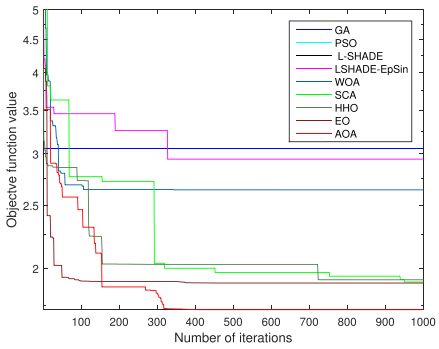


Fig. 6 Exploration and exploitation phases in AOA on the CEC'17 functions with 50 dimensions



(a) Schematic design.



(b) Function values versus number of iterations.

Fig. 7 Welded beam design problem

problem, since then it is used as benchmark for performance evaluation of optimization methods. The cost is optimized subject to shear stress (τ), bending stress (σ) in the beam, buckling load on the bar (P_b), end deflection of the beam (δ), and side constraints. The four decision variables are $h(x_1)$, $l(x_2)$, $t(x_3)$, $b(x_4)$. Appendix A provides mathematical details of the problem.

The results of AOA on welded beam problem are presented in Tables 6 and 7. The best solution obtained by AOA and other counterparts are given in Table 6, whereas

Table 6 Best solution obtained from competitor algorithms for the welded beam problem

Algorithm	h	l	t	b	f_{cost}
GA	0.2372	6.1252	8.1687	0.29	2.6740
PSO	0.2843	7.5333	7.6664	0.3274	3.2731
L-SHADE	0.2389	3.4067	9.6383	0.2901	2.0701
LSHADE-EpSin	0.2884	3.1057	9.3491	0.2999	2.0157
WOA	0.329	2.5471	6.8078	0.3789	2.3584
SCA	0.1947	3.7831	9.1234	0.2077	1.7796
HHO	0.2134	3.5601	8.4629	0.2346	1.8561
EO	0.2057	3.4705	9.0366	0.2057	1.7449
AOA	**0.2057**	**3.4705**	**9.0366**	**0.2057**	**1.7249**

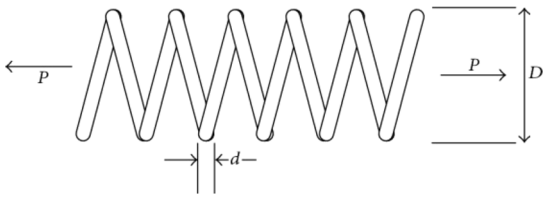
Bold entries highlight the best results achieved by a particular algorithm on a particular problem

Table 7 Results obtained from competitor algorithms for the welded beam problem

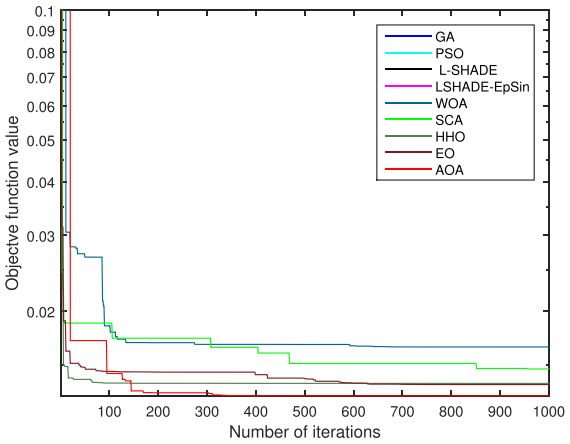
Algorithm	Best	Mean	Worst	Std. Dev.
GA	2.6740	2.5324	2.0410	3.30E+01
PSO	3.27314	3.57351	4.02801	4.00E-01
L-SHADE	2.0701	2.1361	3.2226	5.88E+01
LSHADE-EpSin	2.0157	2.4165	2.8978	4.49E+01
WOA	2.3584	2.5685	2.7862	2.14E-01
SCA	1.7796	1.9326	2.0683	7.40E-02
HHO	1.8561	1.9302	1.9759	6.47E-02
EO	1.7449	1.7555	1.8849	1.86E-03
AOA	1.7249	**1.7304**	1.8716	2.67E-02

Bold entries highlight the best results achieved by a particular algorithm on a particular problem

statistical comparison is presented in Table 7. According to the comparative results, AOA achieved optimum parameter values resulting in best cost function value 1.72485 for this particular design problem. Moreover, AOA also showed better convergence ability than other eight counterparts while solving this problem (Fig. 7b).



(a) Schematic design.



(b) Function values versus number of iterations.

Fig. 8 Tension/compression string design problem

Table 8 Best solution obtained from competitor algorithms for the tension/compression spring problem

Algorithm	d	D	N	f_{cost}
GA	0.0598	0.4121	9.1320	0.019824
PSO	0.0582	0.7952	5.2794	1.733190
L-SHADE	0.0555	0.4706	7.4552	0.018662
LSHADE-EpSin	0.0592	0.4983	8.8980	0.017227
WOA	0.0507	0.3339	12.7645	0.012683
SCA	0.0500	0.3171	14.1417	0.012797
HHO	0.0562	0.4754	6.6670	0.013016
EO	0.0512	0.3445	12.0455	0.012682
AOA	**0.0508**	**0.3348**	**11.7020**	**0.012681**

Bold entries highlight the best results achieved by a particular algorithm on a particular problem

4.3.2 Tension/compression spring design problem

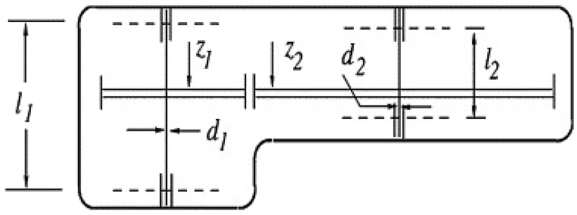
The second problem is to minimize weight of the spring (Fig. 8) based on certain constraints; such as, outside diameter limits, surge frequency, minimum deflection, and shear stress. The detail of this problem is well explained in [48]. The problem consists of three independent variables d or x_1 (wire diameter), D or x_2 (coil diameter), and P or x_3 (number of active coils). The mathematical model of tension/compression spring design problem is provided in Appendix B.

On this problem also, AOA generated best parameter values among other competitive algorithm (Table 8). Table 9 suggests that the objective function value for tension/compression spring design achieved by AOA (0.01268) is lower than the ones achieved by eight other algorithms. The convergence ability of AOA and other algorithms is illustrated via Fig. 8b.

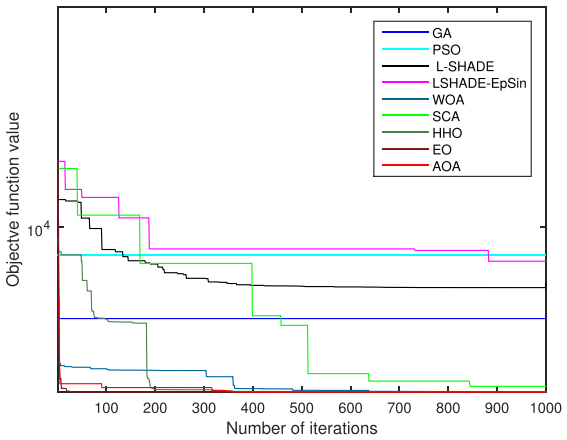
Table 9 Results obtained from competitor algorithms for the tension/compression spring problem

Algorithm	Best	Mean	Worst	Std. Dev.
GA	0.019824	0.028191	0.034820	2.45E+01
PSO	1.733190	1.143211	1.745412	8.48E+05
L-SHADE	0.018662	0.019992	0.022592	1.48E+02
LSHADE-EpSin	0.017237	0.016214	0.020034	1.01E+01
WOA	0.012683	0.014709	0.017211	2.30E-03
SCA	0.012807	0.013859	0.015869	4.30E-04
HHO	0.013026	0.014160	0.016034	1.64E-03
EO	0.012682	0.013536	0.015711	2.20E-04
AOA	0.012681	**0.013369**	0.015625	7.44E-04

Bold entries highlight the best results achieved by a particular algorithm on a particular problem



(a) Schematic design.



(b) Function values versus number of iterations.

Fig. 9 Speed reducer design problem

4.3.3 Speed reducer design problem

The third engineering design optimization problem taken in this study is speed reducer design problem (Fig. 9). Here, an optimization method is to optimize or minimize the weight of the mechanical device based on certain constraints associated with multiple components such as gear teeth bending stress, surface stress, shafts stresses, and transverse deflections of the shafts [49]. The weight is calculated with the help of seven variables, respectively, face width (b or x_1), teeth module (m or x_2), pinion teeth number (z or x_3), first shaft length between bearings (l_1 or x_4), second shaft length between bearings (l_2 or x_5), and first diameter (d_1 or x_6) and second shafts (d_2 or x_7). Appendix C provides mathematical expression of the problem.

The proposed AOA algorithm generated optimum values for the design parameters of speed reducer problem, compared to rest of the methods applied (Table 10). This resulted in best objective design value $2.995E+03$ against the counterpart methods (Table 11). The convergence graph

Table 10 Best solution obtained from the competitor algorithms for the speed reducer problem

Algorithm	X_1	X_2	X_3	X_4	X_5	X_6	X_7	f_{cost}
GA	3.5592	0.7133	19.659	7.9365	8.0197	3.6719	5.3276	3.73E+03
PSO	3.5307	0.7251	23.7382	8.0033	7.9409	3.8991	5.4332	5.60E+03
L-SHADE	3.3626	0.7418	26.2003	7.5158	8.2479	3.8144	5.2271	4.34E+03
LSHADE-EpSin	3.5987	0.7358	19.8452	7.4775	7.9874	3.4014	5.2777	3.31E+03
WOA	3.4975	0.7	17	7.4863	7.8054	3.7245	5.2853	3.68+E02
SCA	3.4935	0.7	17	7.3	7.8	3.3671	5.2798	3.01E+03
HHO	3.4981	0.7	17	7.6398	7.8	3.3582	5.2853	3.00E+03
EO	**3.4976**	**0.7**	**17**	**7.3**	**7.8**	**3.3501**	**5.2857**	**3.00E+03**
AOA	**3.4976**	**0.7**	**17**	**7.3**	**7.8**	**3.3501**	**5.2857**	**3.00E+03**

Bold entries highlight the best results achieved by a particular algorithm on a particular problem

presented in Fig. 9b also shows that AOA converged to a better optimum location than the competitive algorithms.

4.3.4 Pressure vessel design problem

Lastly, the fourth engineering design problem considered in this study is pressure vessel design problem (Fig. 10). The problem is to minimize the cost of pressure vessel design [50]. The cost is optimized with the help of four design variables: T_s or x_1 (shell thickness), T_h or x_2 (head thickness), R or x_3 (inner radius), and L or x_4 (cylinder length). The mathematical model of the speed reducer design problem is described in Appendix D.

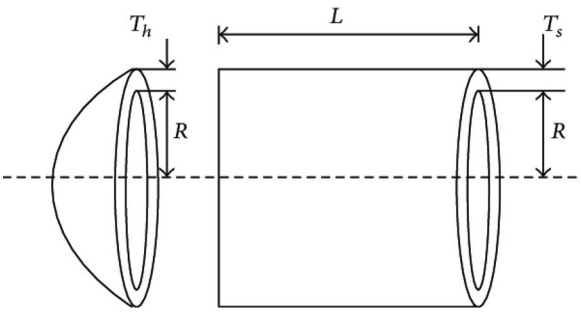
According to the results of pressure vessel design problem (Table 12), the proposed approach AOA achieved parameter settings. The objective function value achieved by AOA was $5.90E+03$ which is better than other algorithms applied on this problem (Table 13). Figure 10b also suggests that AOA maintained efficiency convergence ability on this problem as well.

Table 11 Results obtained from competitor algorithms for the speed reducer problem

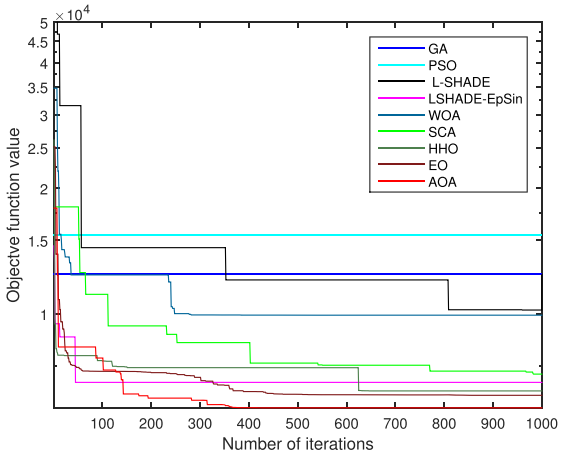
Algorithm	Best	Mean	Worst	Std. Dev.
GA	3.73E+03	8.14E+03	1.73E+04	4.15E+03
PSO	5.60E+03	6.93E+03	8.84E+03	1.70E+03
L-SHADE	4.34E+03	3.18E+04	6.00E+04	2.10E+04
LSHADE-EpSin	3.31E+03	2.28E+04	7.47E+04	2.39E+04
WOA	3.68E+02	4.13E+02	3.68E+03	1.09E+03
SCA	3.01E+03	3.14E+03	3.34E+03	9.03E+01
HHO	3.00E+03	3.89E+03	4.45E+03	7.82E+02
EO	3.00E+03	**3.00E+03**	3.00E+03	1.37E-12
AOA	3.00E+03	**3.00E+03**	3.00E+03	1.22E-12

Bold entries highlight the best results achieved by a particular algorithm on a particular problem

The statistical significance of the proposed method against others in case of engineering problems is reported in Table 14 via p-values generated using nonparametric



(a) Schematic design.



(b) Function values versus number of iterations.

Fig. 10 Pressure vessel design problem

Table 12 Best solution obtained from the competitor algorithms for the pressure vessel design problem

Algorithm	X_1	X_2	X_3	X_4	f_{cost}
GA	0.9187	0.8199	46.3430	167.3579	1.67E+04
PSO	1.6186	0.6296	59.2820	146.9644	1.70E+04
L-SHADE	0.8525	0.5775	56.3105	65.7572	7.67E+03
LSHADE-EpSin	0.9330	0.6982	59.9952	47.5678	6.85E+03
WOA	0.9730	0.6512	50.6804	93.0377	7.11E+03
SCA	0.8951	0.4579	44.8371	147.3388	6.40E+03
HHO	0.9833	0.4758	49.9297	98.9036	6.39E+03
EO	0.7929	0.3914	41.1773	188.3950	5.91E+03
AOA	**0.7900**	**0.3899**	**41.0226**	**190.4405**	**5.90E+03**

Bold entries highlight the best results achieved by a particular algorithm on a particular problem

Table 13 Results obtained from competitor algorithms for the pressure vessel design problem

Algorithm	Best	Mean	Worst	Std. Dev.
GA	1.67E+04	3.63E+04	6.03E+04	1.27E+04
PSO	1.701E+04	2.431E+04	3.559E+04	9.91E+03
L-SHADE	7.67E+03	1.47E+04	3.48E+04	7.74E+03
LSHADE-EpSin	6.85E+03	1.87E+05	1.61E+06	5.00E+05
WOA	7.11E+03	1.05E+04	1.35E+04	3.23E+03
SCA	6.40E+03	7.00E+03	8.22E+03	4.30E+02
HHO	6.39E+03	6.61E+03	6.89E+03	2.54E+02
EO	5.91E+03	6.53E+03	7.30E+03	3.98E+02
AOA	5.90E+03	**6.52E+03**	6.60E+03	4.31E+02

Bold entries highlight the best results achieved by a particular algorithm on a particular problem

Wilcoxon Ranksum test. In Table 14, $\triangle$, $\triangledown$, and $\approx$ indicate that AOA is significantly better than the alternative method, it is significantly inferior than the other, or insignificantly different from the competitive method, respectively. The p-values suggest that AOA produced significantly better results than GA, PSO, L-SHADE, LSHADE-EpSin, WOA, and HHO on all engineering design problems. On the other hand, when compared with SCA results, AOA results were significantly better than SCA for three engineering problems except welded bean design problem. However, AOA remained insignificantly different from EO on all engineering design problems.

5 Conclusion and future works

Generally, different critical considerations regarding the design and performance of an optimization method are simplicity, robustness, and flexibility. Appropriately dealing these features, an optimization algorithm can be made widely acceptable in the research community. In this connection, recently introduced metaheuristic algorithms, especially those inspired by physics, have produced interesting results. This study proposes a new physics-based metaheuristic algorithm that mimics the Archimedes law, called Archimedes optimization algorithm (AOA). In the design of AOA, important criteria related to the simplicity, efficiency, adaptability, and flexibility are effectively ensured. AOA is not only simple, as it holds few control parameters, yet it is robust because the proposed approach can solve optimization problems by generating objective function values with minimum error. The AOA algorithm also maintains ability to adapt its pool of candidate solutions for avoiding trap in the suboptimal locations. The search efficacy of the proposed

Table 14 Wilcoxon ranksum test results for AOA vs. other algorithms (CEC'17 functions with Dim = 50)

Problem	GA	PSO	L-SHADE	LSHADE-EpSin	WOA	SCA	HHO	EO
Tension Spring Design	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	5.57E-10	3.50E-09	1.49E-01
	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\approx$
Welded Beam Design	1.21E-12	3.01E-11	3.01E-11	3.01E-11	2.26E-03	1.09E-01	3.64E-02	1.86E-01
	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\approx$	$\triangle$	$\approx$
Pressure Vessel Design	1.33E-11	1.57E-11	1.89E-11	1.82E-11	1.14E-11	1.52E-11	1.53E-11	6.92E-07
	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$
Speed Reducer Design	3.02E-11	3.02E-11	3.02E-11	6.70E-11	5.49E-11	1.64E-05	8.31E-03	2.84E-01
	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\approx$

approach was tested on complex test functions and four constrained engineering optimization problems. Based on the results, it is evident that AOA not only produced the high quality solutions but also proved its efficiency in finding the global optimum solution faster than several other recently introduced counterparts selected in this study. The AOA algorithm outperformed well-established optimization methods like GA and PSO, state-of-the-art like L-SHADE and LSHADE-EpSin, and other recently introduced methods like WOA, SCA, HHO, and EO. The proposed algorithm exhibited robust search efficiency by balancing exploration and exploitation abilities. The promising solutions on multimodal and composite functions confirmed its exploration, whereas its exploitation was validated by the results of unimodal landscapes.

Future potential research directions related to the proposed AOA involve application on solving more real-world problems. This will help further validate that the algorithm is flexible to generate optimum solutions on a variety of optimization problems.

Acknowledgments The authors would like to thank Halwan University for supporting this research. This research is also partially supported by University of Electronic Science and Technology of China (UESTC) and National Natural Science Foundation of China (NSFC) under Grant No. 61772120.

Compliance with Ethical Standards

Conflict of interest The authors declare that they have no conflict of interest.

Appendix A: Welded beam design problem

Consider $\mathbf{x} = [x_1 x_2 x_3 x_4] = [h\ l\ t\ b]$

Minimize $f(\mathbf{x}) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2)$

Subject to:

$g_1(\mathbf{x}) = \tau(\mathbf{x}) - 13600 \le 0$

$g_2(\mathbf{x}) = \sigma(\mathbf{x}) - 30000 \le 0$

$g_3(\mathbf{x}) = x_1 - x_4 \le 0$

$g_4(\mathbf{x}) = 0.10471 \left(x_1^2\right) + 0.04811 x_3 x_4 (14 + x_2) - 5.0 \le 0$

$g_6(\mathbf{x}) = \delta(\mathbf{x}) - 0.25 \le 0$

$g_7(\mathbf{x}) = 6000 - p_c(\mathbf{x}) \le 0$

where

$$\tau(\mathbf{x}) = \sqrt{(\tau') + (2\tau'\tau'')\frac{x_2}{2R} + (\tau'')^2}$$

$$\tau' = \frac{6000}{\sqrt{2} x_1 x_2}$$

$$\tau'' = \frac{MR}{J}$$

$$M = 6000\left(14 + \frac{x_2}{2}\right)$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}$$

$$j = 2\left\{x_1 x_2 \sqrt{2}\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}$$

$$\sigma(\mathbf{x}) = \frac{504000}{x_4 x_3^2}$$

$$\delta(\mathbf{x}) = \frac{65856000}{\left(30 \times 10^6\right) x_4 x_3^3}$$

$$p_c(\mathbf{x}) = \frac{4.013\left(30 \times 10^6\right)\sqrt{\frac{x_3^2 x_4^6}{36}}}{196}\left(1 - \frac{x_3\sqrt{\frac{30\times 10^6}{4(12\times 10^6)}}}{28}\right)$$

with $0.1 \le x_1, x_4 \le 2.0$ *and* $0.1 \le x_2, x_3 \le 10.0$

Appendix B: Tension/compression spring design problem

Consider:

$\mathbf{x} = [x_1 x_2 x_3] = [d\ D\ N]$

$\text{Min} f(\mathbf{x}) = (x_3 + 2) x_2 x_1^2$

subject to:

$$g_1(\mathbf{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \le 0$$

$$g_2(\mathbf{x}) = \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} - 1 \le 0$$

$$g_3(\mathbf{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0$$

$$g_4(\mathbf{x}) = \frac{x_1 + x_2}{1.5} - 1 \le 0$$

with $0.05 \le x_1 \le 2.0$, $0.25 \le x_2 \le 1.3$, *and* $2.0 \le x_3 \le 15.0$

Appendix C: Speed reducer design problem

$$\text{Min} f(\mathbf{x}) = 0.7854 x_1 x_2^2\left(3.3333 x_3^2 + 14.9334 x_3 - 43.0934\right) - 1.508 x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4 x_6^2 + x_5 x_7^2\right)$$

Subject to:

$$g_1(\mathbf{x}) = \frac{27}{x_1 x_2^2 x_3} - 1 \le 0$$

$$g_2(\mathbf{x}) = \frac{397.5}{x_1 x_2^2 x_3} - 1 \leq 0$$

$$g_3(\mathbf{x}) = \frac{1.93 x_4^3}{x_2 x_3 x_6^4} - 1$$

$$g_4(\mathbf{x}) = \frac{1.93 x_5^3}{x_2 x_3 x_7^4} - 1 \leq 0$$

$$g_5(\mathbf{x}) = \frac{1}{110 x_6^3}\sqrt{\left(\frac{745 x_4}{x_2 x_3}\right)^2 + 16.9 \times 10^6} - 1 \leq 0$$

$$g_6(\mathbf{x}) = \frac{1}{85 x_7^3}\sqrt{\left(\frac{745 x_5}{x_2 x_3}\right)^2 + 157.5 \times 10^6} - 1 \leq 0$$

$$g_7(\mathbf{x}) = \frac{x_2 x_3}{40} - 1 \leq 0$$

$$g_8(\mathbf{x}) = \frac{5 x_2}{x_1} - 1 \leq 0$$

$$g_9(\mathbf{x}) = \frac{x_1}{12 x_2} - 1 \leq 0$$

$$g_{10}(\mathbf{x}) = \frac{1.5 x_6 + 1.9}{x4} - 1 \leq 0$$

$$g_{11}(\mathbf{x}) = \frac{1.1 x_7 + 1.9}{x_5} - 1 \leq 0$$

with $2.6 \leq x_1 \leq 3.6, 0.7 \leq x_2 \leq 0.8, 17 \leq x_3 \leq 28, 7.3 \leq x_4 \leq 8.3, 7.8 \leq x_5 \leq 8.3, 2.9 \leq x_6 \leq 3.9, and\, 5 \leq x_7 \leq 5.5$

Appendix D: Pressure vessel design problem

$$\text{Min}\, f(x) = 0.6224 x_1 x_3 x_4 + 1.7781 x_2 x_3^2 + 3.1661 x_1^2 x_4 + 19.84 x_1^2 x_3$$

Subject to:

$$g_1(x) = -x_1 + 0.0193x$$

$$g_2(x) = -x_2 + 0.00954 x_3 \leq 0$$

$$g_3(x) = -\pi x_3^2 x_4 - (4/3)\pi x_3^3 + 1,296,000 \leq 0$$

$$g_4(x) = x_4 - 240 \leq 0$$

$$0 \leq x_i \leq 100,\ i = 1, 2$$

$$10 \leq x_i \leq 200,\ i = 3, 4$$

References

1. Elaziz MA, Heidari AA, Fujita H, Moayedi H (2020) A competitive chain-based Harris Hawks Optimizer for global optimization and multi-level image thresholding problems. Appl Soft Comput 106347, in press
2. Taradeh M, Mafarja M, Heidari AA, Faris H, Aljarah I, Mirjalili S, Fujita H (2019) An evolutionary gravitational search-based feature selection. Appl Soft Comput 497:219–239
3. Houssein EH, Saad MR, Hashim FA, Shaban H, Hassaballah M (2020) Lévy flight distribution: a new metaheuristic algorithm for solving engineering optimization problems. Eng Appl Artif Intell 94:103731
4. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Future Gener Comput Syst 97:849–872
5. Houssein EH, Hosney ME, Oliva D, Mohamed WM, Hassaballah M (2020) A novel hybrid Harris hawks optimization and support vector machines for drug design and discovery. Comput Chem Eng 133:106656
6. Neggaz N, Houssein EH, Hussain K (2020) An efficient henry gas solubility optimization for feature selection. Exp Syst Appl 113364
7. Hashim FA, Houssein EH, Hussain K, Mabrouk MS, Al-Atabany W (2020) A modified Henry gas solubility optimization for solving motif discovery problem. Neural Comput Appl 32(14):10759–10771
8. Hashim F, Mabrouk MS, Al-Atabany W (2017) GWOMF: Grey Wolf Optimization for motif finding. In: 2017 13th international computer engineering conference (ICENCO), pp 141–146
9. Kar AK (2016) Bio inspired computing—a review of algorithms and scope of applications. Exp Syst Appl 59:20–32
10. Hussain K, Salleh MNM, Cheng S, Shi Y (2019) Metaheuristic research: a comprehensive survey. Artif Intell Rev 52(4):2191–2233
11. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
12. Eberhart RC, Kennedy J (1995) A new optimizer using particle swarm theory. In: MHS'95 Proceedings of the sixth international symposium on micro machine and human science, pp 39–43
13. Holland JH (1992) Genetic algorithms. Sci Am 267(1):66–72
14. Dorigo M, Di Caro G (1999) Ant colony optimization: a new meta-heuristic. In: Proceedings of the 1999 congress on evolutionary computation, 1999 CEC 99, pp 1470–1477
15. Karaboga D, Gorkemli B, Ozturk C, Karaboga N (2014) A comprehensive survey: artificial bee colony (abc) algorithm and applications. Artif Intell Rev 42(1):21–57
16. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
17. Zhao W, Wang L (2016) An effective bacterial foraging optimizer for global optimization. Inf Sci 329:719–735
18. Mirjalili S (2015) Moth-flame optimization algorithm. Knowl-Based Syst 89:228–249
19. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm. Adv Eng Softw 114:163–191
20. Mirjalili SM, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
21. Liu S-H, Mernik M, Hrnčič D, Črepinšek M (2013) A parameter control method of evolutionary algorithms using exploration and exploitation measures with a practical application for fitting sovova's mass transfer model. Appl Soft Comput 13(9):3792–3805
22. Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech 213:267–289
23. Rashedi E, Nezamabadi-pour H, Saryazdi S (2009) Gsa: a gravitational search algorithm. Inf Sci 179(13):2232–2248
24. Kaveh A, Khayatazad M (2012) A new meta-heuristic method: ray optimization. Comput Struct 112:283–294
25. Hashim FA, Houssein EH, Mabrouk MS, Al-Atabany W, Mirjalili S (2019) Henry gas solubility optimization: a novel physics-based algorithm. Future Gener Comput Syst 101:646–667
26. Lam AYS, Li VOK (2010) Chemical-reaction-inspired meta-heuristic for optimization. IEEE Trans Evol Comput 14(3):381–399
27. Mirjalili S (2016) Sca: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96(96):120–133

28. Salimi H (2015) Stochastic fractal search. Knowl-Based Syst 75:1–18
29. Kaveh A, Dadras A (2017) A novel meta-heuristic optimization algorithm: thermal exchange optimization. Adv Eng Softw 110:69–84
30. Hussain S, Ahmad AI, Mutlag AH (2015) Lightning search algorithm. Appl Soft Comput 36:315–333
31. Zhao W, Wang L, Zhang Z (2019) Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. Knowl-Based Syst 89:283–304
32. Faramarzi A, Heidarinejad M, Stephens B, Mirjalili S (2020) Equilibrium optimizer: a novel optimization algorithm. Knowl-Based Syst 191:105190
33. Kaveh A, Share MAM, Moslehi M (2013) Magnetic charged system search: a new meta-heuristic algorithm for optimization. Acta Mech 224(1):85–107
34. Abedinpourshotorban H, Shamsuddin SM, Beheshti Z, Jawawi DNA (2016) Electromagnetic field optimization: a physics-inspired metaheuristic optimization algorithm. Swarm Evol Comput 26:8–22
35. Javidy B, Hatamlou A, Mirjalili S (2015) Ions motion algorithm for solving optimization problems. Appl Soft Comput 32:72–79
36. Mavrovouniotis M, Li C, Yang S (2017) A survey of swarm intelligence for dynamic optimization: algorithms and applications. Swarm Evol Comput 33:1–17
37. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
38. Rorres C (2004) Completing book ii of archimedes's on floating bodies. Math Intell 26(3):32–42
39. Rorres C (2016) Across neighborhood search for numerical optimization. Inf Sci 329:597–618
40. Cheng S, Shi Y, Qin Q, Zhang Q, Bai R (2014) Population diversity maintenance in brain storm optimization algorithm. J Artif Intell Soft Comput Res 4(2):83–97
41. Hussain K, Salleh MNM, Cheng S, Shi Y (2018) On the exploration and exploitation in popular swarm-based metaheuristic algorithms. Neural Comput Appl 31(11):7665–7683
42. Črepinšek M, Liu S-H, Mernik M (2013) Exploration and exploitation in evolutionary algorithms: a survey. ACM Comput Surv 45(3):1–35
43. Črepinšek M, Mernik M, Liu S-H (2011) Analysis of exploration and exploitation in evolutionary algorithms by ancestry trees. Int J Innov Comput Appl 3(1):11–19
44. Cheng S, Lu H, Lei X, Shi Y (2018) A quarter century of particle swarm optimization. Complex Intell Syst 4(3):227–239
45. Eskandar H, Sadollah A, Bahreininejad A, Hamdi M (2012) Water cycle algorithm—a novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput Struct 110(1):151–166
46. Wu G, Mallipeddi R, Suganthan P (2017) Problem definitions and evaluation criteria for the cec 2017 competition on constrained real-parameter optimization, National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, Technical Report. http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2017
47. Coello CAC (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Ind 41(2):113–127
48. Sadollah A, Bahreininejad A, Eskandar H, Hamdi M (2013) Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems. Appl Soft Comput 13(5):2592–2612
49. Mezura-Montes E, Coello CAC (2005) Useful infeasible solutions in engineering optimization with evolutionary algorithms. In: Mexican international conference on artificial intelligence, pp 652–662
50. Kannan B, Kramer SN (1994) An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. J Mech Des 116(2):405–411