Knowledge-Based Systems 260 (2023) 110146

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Fick's Law Algorithm: A physical law-based algorithm for numerical optimization

Fatma A. Hashim [a], Reham R. Mostafa [b], Abdelazim G. Hussien [c,d,*], Seyedali Mirjalili [e,f,g], Karam M. Sallam [h]

[a] Faculty of Engineering, Helwan University, Egypt
[b] Department of Information Systems, Faculty of Computers and Information Sciences, Mansoura University, Mansoura 35516, Egypt
[c] Department of Computer and Information Science, Linköping University, Linköping, Sweden
[d] Faculty of Science, Fayoum University, Egypt
[e] King Abdulaziz University, Jeddah, Saudi Arabia
[f] Centre of Artificial Intelligence Research and Optimisation, Torrens University Australia, Brisbane, Australia
[g] Yonsei Frontier Lab, Yonsei University, Seoul, South Korea
[h] School of IT and Systems, University of Canberra, ACT 2601, Australia

ARTICLE INFO

Article history:
Received 14 August 2022
Received in revised form 14 November 2022
Accepted 19 November 2022
Available online 29 November 2022

Keywords:
Metaheuristic
Optimization
Physics-inspired
Exploration and exploitation
Local optima

ABSTRACT

Recently, many metaheuristic optimization algorithms have been developed to address real-world issues. In this study, a new physics-based metaheuristic called Fick's law optimization (FLA) is presented, in which Fick's first rule of diffusion is utilized. According to Fick's law of diffusion, molecules tend to diffuse from higher to lower concentration areas. Many experimental series are done to test FLA's performance and ability in solving different optimization problems. Firstly, FLA is tested using twenty well-known benchmark functions and thirty CEC2017 test functions. Secondly, five real-world engineering problems are utilized to demonstrate the feasibility of the proposed FLA. The findings are compared with 12 well-known and powerful optimizers. A Wilcoxon rank-sum test is carried out to evaluate the comparable statistical performance of competing algorithms. Results prove that FLA achieves competitive and promising findings, a good convergence curve rate, and a good balance between exploration and exploitation. The source code is currently available for public from: https://se.mathworks.com/matlabcentral/fileexchange/121033-fick-s-law-algorithm-fla.

1. Introduction

Due to the increase in sophistication and difficulty of real-world problems, the need to develop more efficient optimization techniques, especially meta-heuristic optimization algorithms, has increased in recent decades. These methods are mainly stochastic in nature and are used to approximate optimum solutions to various optimization problems [1–5]. Such optimization algorithms supersede conventional optimization algorithms due to gradient-free mechanisms and high local optimal avoidance capabilities [6–10]. In an optimization procedure, the objective function's value is minimized or maximized to identify the best decision variables for a given function or problem [11–13]. From a general perspective, real-world optimization problems include a number of non-linear constraints, intensive computations, large search spaces, and non-convex complexities that render them difficult to solve [14–17].

Recently, meta-heuristic algorithms have gained increasing attention and huge acceptance due to the following benefits (i) flexibility and simplicity; (ii) ease of implementation due to simple ideas; (iii) ability to avoid suboptimal regions; and (iv) no knowledge of the objective function gradient is required. However, metaheuristic algorithms can find not an optimal solution but a near optimal one. Two essential search methods are included in meta-heuristic optimization algorithms: (1) exploitation/diversification and (2) exploration/intensification [18, 19]. Exploration will globally explore the search space. This is linked to local optimum prevention and the resolution of local optima entrapment. On the other hand, exploitation is the capacity to investigate neighboring promising alternative solutions to enhance their quality locally [20–22]. Excellent algorithm success involves a careful balance of these two strategies [23–25]. Each algorithm uses these two features, but with a slightly different set of operators and mechanisms.

* Corresponding author at: Faculty of Science, Fayoum University, Egypt.
E-mail addresses: fatma_hashim@h-eng.helwan.edu.eg (F.A. Hashim), reham_2006@mans.edu.eg (R.R. Mostafa), abdelazim.husien@liu.se, aga08@fayoum.edu.eg (A.G. Hussien), ali.mirjalili@gmail.com (S. Mirjalili), karam.sallam@canberra.edu.au (K.M. Sallam).

https://doi.org/10.1016/j.knosys.2022.110146
0950-7051/

Meta-heuristics are often classified according to their inspiration from evolutionary algorithms, swarm intelligence algorithms, physics-based methods, and human-based methods [26]. The approaches classified as evolutionary algorithms focus on natural evolution concepts and employ operators inspired by biological phenomena such as crossover and mutation. Darwin's theory of evolution served as the inspiration for the genetic algorithm (GA), one of the most famous evolutionary algorithms [27]. This group's other traditional approaches include genetic programming (GP) [28], differential evolution (DE) [29], and evolution strategy [30].

Another meta-heuristic category, which simulates the actions of animals in crowds when moving or hunting [31,32], are swarm intelligence algorithms. This group's key trait is the ongoing process of sharing organism knowledge of all animals via the optimization course. This group's traditional approaches include Bat algorithm [33], Chimp Optimization Algorithm (COA) [34], Slime Mold Algorithm (SMA) [35], Salp Swarm Algorithm (SSA) [36], Harris Hawks Optimization (HHO) [37], Dolphin Echolocation Algorithm (DEA) [38], COOT bird [39], Fox Red Optimization (FRO) [40], Snake Optimizer (SO) [41], Virus Colony Search (VCS) [42], Black Widow Optimization (BWO) [43], Marine Predators Algorithm (MPA) [44], Butterfly Optimization Algorithm (BFA) [45], Symbiotic Organisms Search (SOS) [46], Aquila Optimizer (AO) [47], and Reptile Search Algorithm (RSA) [48].

Besides the previous category of optimization algorithms, there are physics-based approaches. This group originally stems from the physical rules that control the surrounding environment. Simulated Annealing Algorithm (SAA) [49], Vortex Search Algorithm (VSA) [50], Gravitational Search Algorithm (GSA) [51], Multi-verse Optimizer (MVO) [52], Henry Gas Solubility Optimizer (HGSO) [53], Lightning Search Algorithm (LSA) [54], Water Cycle Algorithm (WCA) [55], Light Spectrum Optimizer (LSO) [56] and gradient-based optimizer [57] are all the most widely used algorithms throughout this group.

Human-based methods, inspired by human interactions and behavior in groups, are the optimization algorithm's last category [58]. Imperialist Competitive Algorithm (ICA) [59] is one of the most commonly used algorithms in this group and is based on human socio-political development. The Teaching-Learning-Based Optimization (TLB) algorithm [60] is another algorithm within this category.

A wide range of problems have been addressed by utilizing these various methods, for example; feature selection [61–63], intrusion detection systems [64], image classification [65], task scheduling [66,67], image segmentation [68,69], optimal allocation of power resources [70], optimal power flow [71], identifying photovoltaic models [72,73], medical diagnosis [25], streamflow prediction [74], industrial engineering [75], and others. Regardless of the success of traditional and current optimization algorithms, no algorithm can ensure that the best global optimal solutions for varied optimization problems will be found. The No-Free-Lunch theorem [76] in search and optimization has demonstrated this. This theory inspired us to develop a new optimization algorithm, known as the Fick's Law Algorithm (FLA), that can be more effectively address a variety of optimization issues. FLA algorithm has 3 different phases: diffusion phase, equilibrium phase, and steady state phase.

The main contributions of this paper can be summarized as follows:

- A novel Physics-inspired population algorithm named Fick's Law Algorithm (FLA) is introduced.
- FLA is tested using 20 classical benchmark functions with different dimensions: 30, 50, and 100.
- FLA is tested using 29 CEC 2017 with different dimensions: 10, and 30.

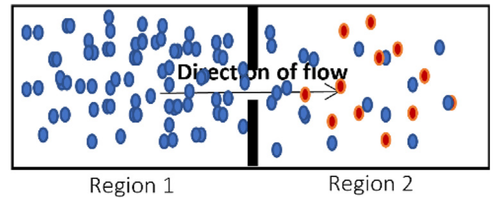


Fig. 1. Directions of flow.

- FLA is tested using five real-world constrained engineering problems.
- FLA is compared with different 12 state-of-art algorithms.

The remainder of the paper is organized as: Section 2 describes the Fick's Law Algorithm developed in this paper whereas Section 3 presents the results, discussion, and evaluation of FLA on various optimization problems. Section 4 presents the conclusion of the current work and recommends future directions.

2. Fick's Law Algorithm

All population-based algorithms try to get the best solution for a specific optimization issue since a single run is not sufficient to guarantee an optimal solution. However, a significant number of random solutions and optimization iterations increase the possibility of obtaining the global optimum solution for the given issue [10,35]. Regardless of the variations between metaheuristic algorithms in population-based optimization techniques, the optimization process is comprised of 2 phases: exploration and exploitation. The first refers to an algorithm's search agents covering a large portion of the search space to avoid local solutions, whereas the exploration phase is the enhancement of solution correctness. In this study, a new physics-based model is developed for optimization purposes. This model is based on the concept of Fick's laws of diffusion. This section consists of two parts: the inspiration for the FLA algorithm and its mathematical model.

FLA is comprised of three different stages, namely: the diffusion phase, the equilibrium phase, and the steady state phase. FLA is thought to be an effective and powerful algorithm because it can find a balance between these phases, which means that exploration and exploitation are both done in a fair way.

2.1. Fick's law

Fick's equations of diffusion are mathematical assertions that describe how particles in random thermal motion tend to move from an area of higher concentration to a region of lower concentration, as demonstrated in 1. The laws also define the relationship between diffusion rate and the three elements that influence diffusion.

It states that 'the rate of diffusion is directly proportional to both the surface area and concentration difference and is inversely proportional to the thickness of the membrane'. A diffusion process that obeys Fick's laws is called normal or Fickian diffusion; otherwise, it is called anomalous diffusion or non-Fickian diffusion.

Three-dimensional diffusion is mathematically defined by Fick's diffusion law, which stipulates that the diffusion flow is proportional to the concentration gradient, as follows:

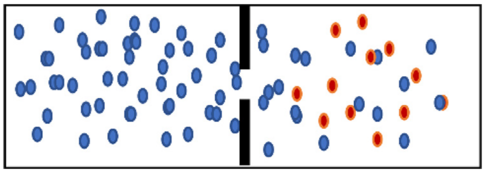


Fig. 2. Diffusion stage in which particles move from one region to another.

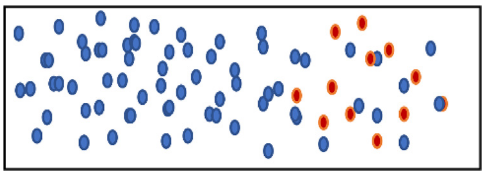


Fig. 3. Equilibrium stage in which the concentration in the 2 regions are nearly equal.

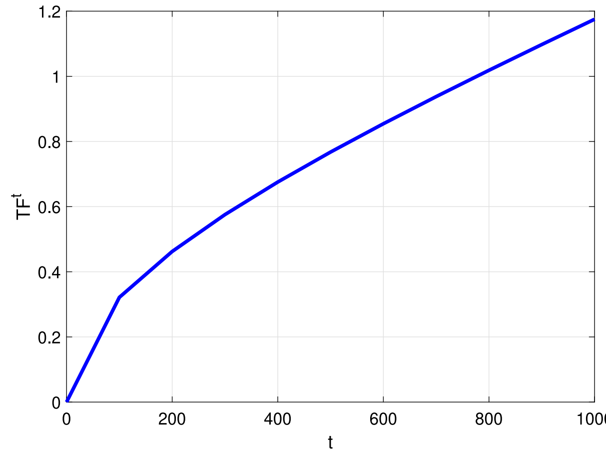


Fig. 4. Proposed non-linear transfer function.

$$F = -D\nabla C \tag{1}$$

where C is the concentration of the diffusing particles, F is the diffusion flux (particles per square meter per second), and D is the diffusion constant, which has units of cm^2 per second. For a one-dimensional problem, Fick's law reduces to:

$$F = -D\frac{dC}{dx} \tag{2}$$

where:

F = diffusion flux (kg/[m^2 s])

D = effective diffusivity (m^2 of coal control surface area/[s]) diffusion flux

dC/dx = concentration gradient ([m^3 of gas]/[m^3 of coal]/[m length along gradient])

C is in [m^3 of gas]/[m^3 of coal].

2.2. Inspiration source

In FLA, we simulate Fick's law to find the stable positions of the molecules. We presented three phases of motion, namely: (1) diffusion operator, (2) equilibrium operator (EO), and (3) steady-state operator (SSO).

In the first phase, we consider that at the beginning of the experiment there are two regions and there is a high difference in concentration between them; this leads to the movement of particles from one region to another according to the concentration difference; we called this phase diffusion operator (DO); in Fig. 2, we consider two regions, one with high concentration (region 1) and the other with low concentration (region 2). According to flicks law, the particles will move from region 1 to region 2.

After that, the second phase comes; we called it the equilibrium operator. In this phase, the two concentrations are nearly equal, and the particles try to reach the equilibrium case. This occurs by the movement of each particle in a new region (without migration to another region) according to the best stable position in this region. as presented in Fig. 3.

To transfer the algorithm to different stages that confirm the success of the balance between exploration and exploitation and avoid stagnation at the local optimum, the third phase is what we call the steady-state operator (SSO); After the two concentrations in two regions reach equilibrium, we will move the barrier, and the gab due to the barrier will cause the agents to be more stable by moving to the most stable location in the tank.

The mathematical formulation of FLA is detailed in the following subsections.

2.3. FLA mathematical model

This subsection presents the mathematical model and the main steps of the FLA algorithm.

Step 1: initialization In FLA, the optimization process begins with a set of candidate solutions (X) as shown in the matrix in Eq. (3), which is generated randomly, and the best candidate solution in each iteration is considered as the best-obtained solution or nearly the optimum so far.

$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,D-1} & x_{1,D} \\ x_{2,1} & \cdots & x_{2,j} & \cdots & x_{2,D-1} & x_{2,D} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N-1,1} & \cdots & x_{N-1,j} & \cdots & x_{N-1,D-1} & x_{N-1,D} \\ x_{N,1} & \cdots & x_{N,j} & \cdots & x_{N,D-1} & x_{N,D} \end{bmatrix} \tag{3}$$

where N is the number of solutions or population size, D is the problem dimension or number of decision variables and j represents the jth decision variable.

Step 2: Clustering

Divide the population into two equal groups N_1, and N_2.

Step 3: Transfer function (TF)

The transfer from exploration to exploitation and vice versa is the main core of the success of any optimization algorithm, Transition Function (TF) is presented to solve this problem. The most popular transfer parameter is the linear parameter that failed in a lot of problems to change from exploratory to exploitative behaviors. In this paper, we propose a nonlinear transfer function that is used as a base to move from exploitation to exploration and vice versa. This TF is described by Eq. (4) and graphically be Fig. 4.

$$TF^t = sinh(t/T)^{c_1} \tag{4}$$

where t represents the iteration number, T is the total number of iterations, and $C_1 = 0.5$.

Step 4: update molecule position: We propose three stages of transfer operators including DO, EO, and SSO. To transfer between three stages, the following equation is used:

$$X_i^t = \begin{cases} DO & TF^t < .9 \\ EO & TF^t \le 1 \\ SSO & TF^t > 1 \end{cases} \tag{5}$$

2.4. Diffusion Operator (DO) (exploration phase)

At the beginning, the two regions have a high difference in concentration, this leads to the transfer of molecules from one region to another according to the concentration of the given region. We propose parameter (T^t_{DO}) which can be given by:

$$T^t_{DO} = C_5 \times TF^t - r \tag{6}$$

$$C_5 = 2 \tag{7}$$

Based on T^t_{DO} value the direction of flow is determined as:

$$X^t_{p,i} = \begin{cases} \text{from i s to j region} & T^t_{D0} < \text{ rand} \\ \text{from j s to i region} & \text{otherwise} \end{cases} \tag{8}$$

Firstly consider that region i has a higher concentration than region j. According to this, some molecules travel from the i's region to the j's one and the remaining molecules in the i's region will be affected by this event. The number of molecules travels from i's region to j's one is given the following:

$$\mathrm{NT_{ij}} \approx \mathrm{N_i} \times \mathrm{r_1} \times (\mathrm{C_4} - \mathrm{C_3}) + \mathrm{N_i} \times \mathrm{C_3} \tag{9}$$

where NT_{ij} refers to the number of molecules that will transfer from the i's group to the j's group, C_3 and C_4 are constants that equal 0.1 and 0.2 respectively.

The number of staying molecules in i's group can be obtained using the following equation:

$$NR_i \approx N_i - NT_{ij} \tag{10}$$

Molecules $\mathrm{NT_{ij}}$ will travel to another region and their positions will be updated mainly on the best equilibrium molecule in region j using:

$$\mathrm{X^{t+1}_{p,i}} = \mathrm{X^t_{EO,j}} + \mathrm{DF^t_{p,i}} \times \mathrm{DOF} \times \mathrm{r_2} \times \left(J^t_{\mathrm{i,j}} \times \mathrm{X^t_{EO,j}} - \mathrm{X^{t+1}_{p,i}}\right) \tag{11}$$

where $\boldsymbol{X}^t_{EO,j}$ is the equilibrium position in region j, DF^t_i is the direction factor which equals either $\{-1,1\}$ that changes randomly and will give high scanning opportunity the given search area and escaping from local optimum, r_2 is a random number $\in [0, 1]$ and DOF is the direction of flow that changes with time and can be given by the following equation:

$$\mathrm{DOF} = \exp\left(-\mathrm{C_2}\left(\mathrm{TF^t} - \mathrm{r_1}\right)\right), \mathrm{C_2} = 2 \tag{12}$$

$J^t_{i,j}$ is diffusion flux and is given by:

$$J^t_{i,j} = -D\frac{dc^t_{i,j}}{dx^t_{i,j}} \tag{13}$$

where D refers to the effective diffusivity constant which equals 0.1, $\frac{\mathrm{dc^t_{i,j}}}{\mathrm{dx^t_{i,j}}}$ refers to the concentration gradient, and $\mathrm{dc}^t_{i,j}$ & $\mathrm{dx^t_{i,j}}$ can be given by the following equations:

$$\mathrm{dc}^t_{i,j} = \mathrm{X^t_{m,j}} - \mathrm{X^t_{m,i}} \tag{14}$$

$$\mathrm{dx^t_{i,j}} = \sqrt{\left(\mathrm{X^t_{EO,j}}\right)^2 - \left(\mathrm{X^t_{p,i}}\right)^2 + \mathrm{eps}} \tag{15}$$

where $\mathrm{X^t_{m,j}}$ and $X^t_{m,i}$ are the mean of molecule position in regions j and i respectively.

The molecules in i's group (NRi) that still existed in region i update their position according to navigation between different three stages: (1) the equilibrium position in region i, (2) the equilibrium position in region i and the boundary of the problem, or (3) no change in their position. The following equation is used to simulate this strategy:

$$\mathrm{X^{t+1}_{p,i}} = \begin{cases} \mathrm{X^t_{EO,i}} & \text{rand } < .8 \\ \mathrm{X^t_{EO,i}} + \mathrm{DOF} \times (\mathrm{r_3} \times (\mathrm{U} - \mathrm{L}) + \mathrm{L}) & \text{rand } < .9 \\ \mathrm{X^{t+1}_{p,i}} & \text{otherwise} \end{cases} \tag{16}$$

where $\mathrm{X^t_{EO,i}}$ is the equilibrium position in region i, U&L are the upper and lower boundaries of the problem, and r_3 is a random number in the interval [0, 1].

For j's molecules, they change their position in the same region without traveling as j region is higher in concentration so they update their position according to the equilibrium position in region j and the boundary of the problem used can be calculated from the following equation:

$$\mathrm{X^{t+1}_{p,j}} = \mathrm{X^t_{EO,j}} + \mathrm{DOF} \times (\mathrm{r_4} \times (\mathrm{U} - \mathrm{L}) + \mathrm{L}) \tag{17}$$

The above strategy will be considered if region i is lower in concentration and if region j is higher, then the vice versa will be occur, where the molecules travels from j's region to i's region

2.5. Equilibrium operator (EO) (transfer phase from exploration to exploitation)

This phase is considered a transfer phase from exploration to exploitation. The molecules update their position using:

$$X^{t+1}_{p,g} = X^t_{EO,p} + Q^t_{EO,g} \times X^t_{p,g} + Q^t_{EO,g} \times \left(MS^t_{p,EO} \times X^t_{EO,g} - X^t_{p,g}\right) \tag{18}$$

where $X^t_{p,g}$ is the position of particle p in group g, $X^t_{EO,g}$ is the equilibrium location in group g, $Q^t_{EO,g}$, $MS^t_{p,EO}$ can be given by the equations Eqs. (19) and (25):

$$Q^t_{EO,g} = R^t_1 \times DF^t_g \times DRF^t_{EO,g} \tag{19}$$

where $Q^t_{EO,g}$ refers to the relative quantity of the region in group g and $DRF^t_{EO,g}$ is the Diffusion rate factor in group g and can be calculated as follows:

$$DRF^t_{EO,g} = \exp\left(-J^t_{p,EO}/TF^t\right) \tag{20}$$

where $J^t_{p,EO}$ can be calculated using the following equation:

$$J^t_{p,EO} = -D\frac{dc^t_{g,EO}}{dx^t_{p,EO}} \tag{21}$$

$dc^t_{g,EO}$, $dx^t_{p,EO}$ can be calculated as follows

$$dc^t_{g,EO} = X^t_{g,EO} - X^t_{m,g} \tag{22}$$

$$dx^t_{p,EO} = \sqrt{\left(X^t_{g,EO}\right)^2 - \left(X^t_{p,g}\right)^2 + eps} \tag{23}$$

$$DF^t_g = \pm 1 \text{ direction factor} \tag{24}$$

$$MS^t_{p,EO} = \exp\left(-\frac{FS^t_{g,EO}}{\left(FS^t_{p,g} + eps\right)}\right) \text{ motion step} \tag{25}$$

$$R^t_1 = \text{rand}[0, 1]_d \quad d = 1: \text{ dimension} \tag{26}$$

where $\mathrm{FS^t_{g,EO}}$ is the best fitness score in group g at time t and $\mathrm{FS^t_{i,g}}$ is the fitness score of particle p in group g at time t.

2.6. Steady state operator (SSO) (exploitation phase)

The final stage in the optimization search is the exploitation or stability phase and the molecules update their position using:

$$X^{t+1}_{p,g} = X^t_{SS} + Q^t_g \times X^t_{p,g} + Q^t_g \times \left(MS^t_{p,g} \times X^t_{SS} - X^t_{p,g}\right) \tag{27}$$

where X^t_{SS} is the steady state location, $X^t_{p,g}$ is the position of particle p.

Q^t_g&$MS^t_{p,g}$ refer to the relative quantity of the region g and the motion step. They can be calculated using Eqs. (28) and (30)

$$Q^t_g = R^t_1 \times DF^t_g \times DRF^t_g \tag{28}$$

where DF^t_g is the direction factor that equals ± 1, R^t_1 is a random number in the interval [0, 1], and DRF^t_g refers to the diffusion rate factor and can be calculated as follows:

$$DRF_{g}^{t} = \exp\left(-J_{p,ss}^{t}/TF^{t}\right) \tag{29}$$

$$MS_{p,g}^{t} = \exp\left(-\frac{FS_{ss}^{t}}{\left(FS_{p,g}^{t} + eps\right)}\right) \tag{30}$$

$J_{p,ss}^{t}$ can be calculated from the following equation:

$$J_{p,ss}^{t} = -D\frac{dc_{g,ss}^{t}}{dx_{p,ss}^{t}} \tag{31}$$

where $dc_{g,ss}^{t}$ and $dx_{p,ss}^{t}$ can be obtained from the following 2 equations:

$$dc_{g,ss}^{t} = X_{m,g}^{t} - X_{ss}^{t} \tag{32}$$

$$dx_{p,ss}^{t} = \sqrt{\left(X_{ss}^{t}\right)^2 - \left(X_{p,g}^{t}\right)^2 + eps} \tag{33}$$

2.7. Balancing between exploration & exploitation

Balancing between exploration and exploitation is one of the key factors of designing any algorithm. FLA has 3 different stages: diffusion stage, equilibrium stage, and steady-state stage. Exploration stage is simulated in diffusion where the molecules will explore the whole search space. After that the transition between exploration & exploitation is done in equilibrium phase. Finally the molecules will exploit the promising areas in steady state phase.

2.8. Pseudo-code of the Fick's law algorithm (FLA)

In this section, the steps of the proposed FLA algorithm are explained and presented in Algorithm 1.

2.9. The computational complexity of FLA

FLA time complexity depends on 3 different factors namely: initialization, updating positions, and evaluation. Let N is the number of individuals, so $O(N)$ is the initialization phase complexity. On the other hand, the complexity of updating an individual is $O(N \times T) + O(N \times T \times Dim)$ where T refers to the maximum iteration number and Dim refers to the number of dimensions. The complexity of evaluating solutions relies on the problem, so it will not be calculated. Lastly, the complexity of FLA is $O(N) + O(N \times T \times Dim) = O(N \times (T \times Dim + 1))$.

2.10. Parameters tuning

Since metaheuristics algorithms will find different solutions in each run due to its stochastic behavior in the nature. If we carry out these experiments many times with full consideration to fractional factorial and full factorial design to find the best combination of all design variables, it will be infeasible for initial study length like this one. Here, we make a configuration for our variables $C1$ to $C4$ using 6 functions ($F3, F5, F8, F11, F15, F21, F25$) from IEEE CEC 2017.

The chosen functions are from different categories (unimodal, multimodal, hybrid, and composite). The four variables are in the following range $C1 \in \{0.5, 1\}$, $C2 \in \{0.5, 1, 1.5, 2, 2.5\}$, $C3 \in \{0.1, 0.2\}$, $C4 \in \{0.1, 0.2\}$, and $C5 \in \{0.5, 1, 2\}$.

We consider the following things:

(1) $C3$ & $C4$ are percentage so they take small number less than 1. take the difference between them so they cannot be equal.

(2) $C1$ is the power of the time operator that transfer between different stages, so it may 1 or less than one.

Algorithm 1 FLA algorithm

Initialization Phase;
  Initialize parameters ($D, C_1, C_2, C_3, C_4, C_5$);
  Initialize the population $X_i (i = 1, 2, \ldots N)$ randomly;
Clustering: Divide population into two equal groups N_1, and N_2;
for s=1:2 **do**
  Compute the fitness function for each molecule in group N_s;
  Find the best molecule in each group and the global optimum;
end for
while $FES \leq MAX_{FES}$ **do**
  if TF < 0.9 **then** %Steady State Operator (SSO)
    for $op = 1 : n_{op}$ **do**
      Calculate Diffusion rate factor using Eq. (29)
      Calculate Motion Step factor using Eq. (30)
      Update individual position using Eq. (27)
    end for
  else if (TF < rand) **then** %Equilibrium Operator (EO)
    for $op = 1 : n_{op}$ **do**
      Calculate Diffusion rate factor using Eq. (20)
      Calculate group relative Quantity using Eq.(19)
      Update individual position using Eq.(18)
    end for
  else %Diffusion Operator (EO)
    Calculate direction of flow using Eq.(12)
    Determine number of molecules that will travel to region using Eq.(9)
    Update individual position using Eq. (11);
    Update Other molecules in region i using Eq. (16);
    Update molecules in region j using Eq. (17);
    Update $FES \leftarrow FES + NP$;
  end if
end while
Return best solution;

(3) $C2$ is used in the direction of flow in Eq. (12), it may be take different values less or more than 1.

(4) C5 used to transfer within different cases in Diffusion Operator (DO), it may be take different values less or more than 1.

From Table 1, It is noticed that the best parameters are $C1 = 1$, $C2 = 2$, $C3 = 0.1$, $C4 = 0.2$, and $C5 = 2$.

3. Experimental results and discussions

The performance of the proposed FLA algorithm is tested in this section by means of various test benchmark functions and real-world optimization problems. Two sets of well-known functions, comprising twenty traditional functions and twenty-nine CEC2017 benchmarks, are used for the numerical validation stage. As examples of real-world applications, two engineering optimization tasks are employed. The FLA is designed to solve the considered test benchmarks and engineering applications for 1000 iterations with 30 search agents. Moreover, FLA has been executed 30 independent times to assess FLA's consistency and reliability, and the average of the results (Average) and standard deviation (STD) have been reported.

The following metaheuristic algorithms have been compared to substantiate the FLA's quality:

- LSHADE-EpSin [77]
- Gravitational Search Algorithm (GSA) [51]
- Sine Cosine Algorithm (SCA) [78]

Table 1
Sensitivity analysis for FLA parameters under different scenarios.

	C1	C2	C3	C4	C5	F3	F5	F8	F11	F15	F21	F25
1	0.5	0.5	0.2	0.1	0.5	26 805.06	644.2756	933.0158	1362.048	310 756.8	2429.529	2960.816
2	0.5	0.5	0.2	0.1	1	26 094.37	647.0744	946.2439	1360.002	605 611.3	2433.827	2977.104
3	0.5	0.5	0.2	0.1	2	31 325.35	655.7939	938.4101	1376.028	322 872.2	2419.905	2956.576
4	0.5	1	0.1	0.2	0.5	30 606.7	637.3561	926.1540	1417.654	235 526.6	2392.93	2944.404
5	0.5	1	0.1	0.2	1	19 795.37	642.4015	932.1042	1360.839	381 642.8	2414.478	2927.799
6	0.5	1	0.1	0.2	2	23 379.11	630.2585	928.3503	1412.218	564 359.1	2435.301	2933.308
7	0.5	1.5	0.2	0.1	0.5	18 773.4	618.3081	909.5405	1335.729	210 736.6	2428.2	2903.431
8	0.5	1.5	0.2	0.1	1	21 785.4	623.1764	909.9823	1247.774	87 008.39	2420.324	2908.238
9	0.5	1.5	0.2	0.1	2	21 292.59	619.5697	903.7490	1238.868	17 247.21	2418.337	2908.483
10	0.5	2	0.1	0.2	0.5	19 055.42	624.1640	905.6163	1198.685	7149.777	2398.162	2907.148
11	0.5	2	0.1	0.2	1	19 940	615.5428	904.7726	1214.947	9115.284	2424.328	2916.775
12	0.5	2	0.1	0.2	2	19 602.29	608.9723	906.3409	1215.079	10 034.77	2399.962	2901.556
13	0.5	2.5	0.2	0.1	1	20 205.23	627.6855	909.7696	1220.223	6007.633	2428.675	2920.526
14	0.5	2.5	0.2	0.1	2	20 346.66	636.3450	915.2494	1233.618	5503.837	2427.675	2932.784
15	1	0.5	0.2	0.1	0.5	40 515.9	654.4465	950.9297	1381.814	1 309 321	2438.02	2965.419
16	1	0.5	0.2	0.1	1	31 678.41	667.5362	947.3037	1356.682	656 056.5	2445.662	3010.949
17	1	0.5	0.2	0.1	2	27 602.35	661.9698	948.1952	1418.294	211 969.9	2452.955	2943.755
18	1	1	0.1	0.2	0.5	20 835.75	621.7912	904.3687	1311.486	177 050.8	2402.733	2944.173
19	1	1	0.1	0.2	1	26 722.56	650.6866	944.2497	1344.467	170 032.9	2416.294	2993.999
20	1	1	0.1	0.2	2	26 121.66	652.6953	943.2163	1372.31	604 266.8	2440.322	2916.758
21	1	1.5	0.2	0.1	2	26 225.65	625.0258	914.5357	1295.642	127 603.8	2417.011	2946.244
22	1	2	0.1	0.2	1	22 617.28	609.4923	907.5939	1214.12	7568.303	2420.843	2928.958
23	1	2	0.1	0.2	2	16 220.03	605.8189	898.7307	1238.726	2740.494	2371.516	2896.496
24	1	2.5	0.2	0.1	2	29 205.55	612.3341	908.4836	1228.033	4306.566	2419.314	2925.598

- Whale Optimization Algorithm (WOA) [79]
- Salp Swarm Algorithm (SSA) [80]
- Thermal Exchange Optimization (TEO) [81]
- Henry Gas Solubility Optimization (HGSO) [53]
- Harris Hawks Optimization (HHO) [37]
- Artificial ecosystem-based optimization (AEO) [82]
- Hunger Games Search (HGS) [83]

The under-consideration algorithms were constructed with the same number of iterations and population size of FLA 1000 and 30, respectively, to provide a fair comparison, resulting in a total of 30 000 function evaluations. Table 2 lists the values utilized for the principal control parameters for the comparison algorithms.

3.1. Standard benchmark functions analysis

In this subsection, the experiments are carried out on different dimensions: 30 dimensions, 50 dimensions, and 100 dimensions for unimodal and multimodal functions. Also, fixed dimensions multi-modal problems with dimensions equal to 2.

3.1.1. Benchmark functions description

Twenty benchmark functions were used to evaluate the FLA's ability in search space exploration, global solutions exploitation, and escaping from local minima. Three kinds of benchmark functions were utilized for assessing the effectiveness of FLA: (1) unimodal, (2) multimodal, and (3) fixed-dimension multimodal functions. Table 3 presents detailed descriptions of these functions. The function dimension, search boundary, and cost function are indicated by *Dim*, *R* and *fmin*(x) respectively.

Three types of testbed problems are used to assess the particular performance of competitive algorithms; unimodal functions (F1–F7) have just one global optimum (no local optimum) and therefore are applied to evaluate the exploitation capacity of the FLA. Whereas multimodal functions (F8–F11), which have numerous local optima, are mostly used to gauge the exploratory capability of the FLA. Due to many local optima in fixed-dimension multimodal functions (F12–F20), they are used to evaluate FLA's capacity to avoid local optima and the extent to which exploration and exploitation are balanced.

The FLA has been evaluated with these benchmarks and compared with these well-regarded algorithms such as LSHADE-EpSin, GSA, SCA, WOA, SSA, TEO, HGSO, HHO, AEO, and HGS. To have a fair comparison, all algorithms were run independently 30 times and each algorithm starts with a population of 30 search agents randomly generated.

3.1.2. Results for 30D

The detailed results (Best, Worst, Median, Average, and Std) are presented in Tables (S1 & S2) in the supplementary material file. The summary of the results obtained from the proposed FLA algorithm and the rival algorithms is given in Table 4. It is clear from Table 4 that the proposed FLA algorithm is superior to all competing algorithms for both best and average obtained results in all functions. In regards to best results, it can be seen that the proposed FLA algorithm obtains better results than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 1, 11, 7, 0, 8, 11, 9, 11, 11 and 11 problems, respectively. The proposed algorithm is worse than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 1, 0, 0, 1, 0, 0, 0, 0, 0, 0 test problems, respectively. In regards to the average results as it can be seen in Table 4, FLA obtains superior results to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpS in for 6, 11, 7, 0, 8, 11, 9, 11, 11 and 11 problems, respectively. The proposed FLA is inferior to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 0, 0, 0, 1, 0, 0, 0, 0, 0, and 0 test functions, respectively.

The Wilcoxon test is performed to determine the significant difference between two competing algorithms with the results presented in Table 4. It is concluded from Table 4 that, the proposed FLA is significantly better than all the rival algorithms for both best and average results, which is confirmed by the *P*-value column in Table 4. As a further analysis, the Friedman rank test is conducted to rank all the rival algorithms, with the results presented in Table 5. It is clear from Table 5 that the proposed FLA algorithm comes in the second rank for Best results after HGSO algorithm, and in the first rank with HGSO for average results.

3.1.3. Results for 50D

The detailed results (Best, Worst, Median, Average, and Std) are presented in tables (S3 & S4) in the supplementary material

Table 2
Parameter settings.

Algorithm	Parameters
LSHADE-EpSin	Pbest = 0.1 Arc rate = 2
GSA	Acceleration coefficient ($\alpha = 20$) Initial gravitational constant (G0 = 100)
SCA	$a = 2$
WOA	a variable decreases linearly from 2 to 0 (Default) $a2$ linearly decreases from 1 to 2 (Default)
SSA	$c_2 = rand$ $c_3 = rand$
TEO	$u = 1$ v = 0.001
HGSO	Gases number = 50 Cluster number = 5 M1 and M2 = 0.1 and 0.2 l_1, l_2, l_3 are constants with values equal 5E − 03, 100, 1E − 02 (fixed for benchmark functions) l_1, l_2, l_3 are constants with values equal 1, 10, 1 (fixed for engineering problems) $\beta = 1, \alpha = 1$ and $K = 1$
HHO	$\beta = 1.5$ $E_0 \in [-1, 1]$
AEO	$r, r_1, r_2 = rand$ $h = 2 \times rand$
HGS	$k = 0.03$ $r_1, r_2, r_3, r_4, r_5 = rand$
FFA	$w = 1$ $Q = 0.6$ $\beta = 0.8$ $\alpha = 0.9$
AVOA	$L_1 = 0.8$ $L_2 = 0.2$ $K = 2.5$ $P_1 = 0.6$ $P_2 = 0.4$ $P_3 = 0.6$
GTO	$\beta = 3$ $w = 0.8$ $\alpha = 0.9$

file. The summary of the results obtained from the proposed FLA algorithm and the rival algorithms is given in Table 6. It is clear from Table 6 that the proposed FLA algorithm is superior to all competing algorithms for both best and average obtained results. In regards to best results, it can be seen that the proposed FLA algorithm obtains better results than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 1, 11, 7, 0, 8, 11, 9, 11, 11 and 11 problems, respectively. The proposed algorithm is worse than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 1, 0, 0, 1, 0, 0, 0, 0, 0, 0 test problems, respectively. In regards to the average results as it can be seen in Table 6, FLA obtains superior results to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpS in for 5, 11, 7, 0, 8, 11, 9, 11, 11 and 11 problems, respectively. The proposed FLA is inferior to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 1, 0, 0, 1, 0, 0, 0, 0, 0, and 0 test functions, respectively.

The Wilcoxon test is performed to determine the significant difference between two competing algorithms with the results presented in Table 6. It is concluded from Table 6 that, the proposed FLA is significantly better than all the rival algorithms for both best and average results, which is confirmed by the P-value column in Table 6. As a further analysis, the Friedman rank test is conducted to rank all the rival algorithms, with the results presented in Table 7. It is clear from Table 7 that that proposed FLA algorithm comes in the first rank with HGSO algorithm for both Best and Average results followed by HGS which comes in second place for both Best and Average results.

3.1.4. Results for 100D

The detailed results (Best, Worst, Median, Average, and Std) are presented in Tables (S5 & S6) in the supplementary material file. The summary of the results obtained from the proposed FLA algorithm and the rival algorithms is given in Table 8. It is clear from Table 8 that the proposed FLA algorithm is superior to all competing algorithms for both best and average obtained results. In regards to best results, it can be seen that the proposed FLA algorithm obtains better results than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 1, 11, 7, 0, 7, 11, 9, 11, 11 and 11 problems, respectively. The proposed algorithm is worse than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 1, 0, 0, 1, 1, 0, 0, 0, 0, 0 test problems, respectively. In regards to the average results as it can be seen in 8, FLA obtains superior results to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpS in for 6, 11, 7, 0, 7, 11, 9, 11, 11 and 11 problems, respectively. The proposed FLA is inferior to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 1, 0, 0, 2, 1, 0, 0, 0, 0, and 0 test functions, respectively.

The Wilcoxon test is performed to determine the significant difference between two competing algorithms with the results presented in Table 8. It is concluded from Table 8 that the proposed FLA is significantly better than all the rival algorithms for both best and average results, which is confirmed by the P-value column in Table 8. As a further analysis, the Friedman rank test is conducted to rank all the rival algorithms, with the results presented in Table 9. It is clear from Table 9 that the proposed

Table 3
Description of benchmark functions.

#	Expression	Name	Dim	R	fmin(x)
Unimodal functions					
1	$f_{\text{Chung Reynolds}}(x) = \left(\sum_{i=1}^{n} x_i^2\right)^2$	Chung Reynolds	30, 50, 100	[−100,100]	0
2	$f_{\text{sphere}}(x) = \sum_{i=1}^{ll} x_i^2$	De Jong's (sphere)	30, 50, 100	[−5.12, 5.12]	0
3	$f_{\text{POWELL Singular 2}}(x) = \sum_{j=2}^{D-2} (x_{i-1} - 10x_i)^2 + 5(x_{i+1} - x_{i+2})^2 + (x_i - 2x_{i+1})^4 + 10(x_{i-1} - x_{i+2})^4$	POWELL Singular	30, 50, 100	[−4,5]	0
4	$f_{\text{Powell Sum}} = \sum_{i=1}^{n} \lvert x_i \rvert^{i+1}$	Powell Sum	30, 50, 100	[−1,1]	0
5	$f_{\text{Schwefel 2.22}} = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	Schwefel 2.22	30, 50, 100	[−100,100]	0
6	$f_{\text{Schwefel2.23}} = \sum_{i=1}^{n} x_i^{10}$	Schwefel 2.23	30	[−10,10]	0
7	$f_{\text{Sum squares}}(x) = \sum_{i=1}^{n} i * x_i^2$	Sum Squares	30, 50, 100	[−10,10]	0
Multimodal functions					
8	$f_{\text{Brown}}(x) = \sum_{i=1}^{n-1}\left(\left(x_i^2\right)^{\left(x_{i+1}^2+1\right)} + \left(x_{i+1}^2\right)^{\left(x_i^2+1\right)}\right)$	Brown	30, 50, 100	[−1,4]	0
9	$f_{\text{cigar}}(x) = x_1^2 + 10^6 \sum_{i=2}^{n} x_i^2$	Cigar*	30, 50, 100	[−100,100]	0
10	$f_{\text{Csendes}}(x) = \sum_{i=1}^{n} x_i^6 \left(2 + \sin \frac{1}{x_i}\right)$	Csendes	30, 50, 100	[−1,1]	0
11	$f_{\text{RASTRIGIN}}(x) = 10\,d + \sum_{i=1}^{d}\left[x_i^2 - 10\cos(2\pi x_i)\right]$	Rastrigin **	30, 50, 100	[−5.12, 5.12]	0
fixed-dimension multimodal functions					
12	$f_{\text{Bartels}}(x) = \lvert X_1^2 + X_2^2 + X_1 * X_2 \rvert + \lvert \text{SIN}(X_1) \rvert + \lvert \text{COS}(X_2) \rvert$	Bartels Conn	2	[−500,500]	1
13	$f_{\text{Bohachevsky 1}}(x) = X_1^2 + 2X_2^2 - .3\,\text{COS}(3\Pi X_1) - .4\,\text{COS}(4\Pi X_2) + .7$	Bohachevsky 1	2	[−100,100]	0
14	$f_{\text{Bohachevsky2}}(x) = X_1^2 + 2X_2^2 - .3\,\text{COS}(3\Pi X_1) * .4\,\text{COS}(4\Pi X_2) + .3$	Bohachevsky 2	2	[−100,100]	0
15	$f_{\text{Bohachevsky3}}(x) = X_1^2 + 2X_2^2 - .3\,\text{COS}(3\Pi X_1 + 4\Pi X_2) + .3$	Bohachevsky 3	2	[−100,100]	0
16	$f_{\text{Camel}}(x) = 2x_1^2 - 1.05x_1^4 + \frac{x_1^6}{6} + x_1 x_2 + x_2^2$	Camel– Three Hump	2	[−5,5]	0
17	$f_{\text{Egg Crate}}(x) = x_1^2 + x_2^2 + 25\left(\sin^2(x_1) + \sin^2(x_2)\right)$	Egg Crate	2	[−5,5]	0
18	$f_{\text{Matyas}}(x) = 0.26\left(x_1^2 + x_2^2\right) - .048x_1x_2$	Matyas	2	[−10,10]	0
19	$f_{\text{Sawtoothxy}}(x) = g(r) \cdot h(t)$ where $g(r) = \left[\sin(r) - \frac{\sin(2r)}{2} + \frac{\sin(3r)}{3} - \frac{\sin(4r)}{4} + 4\right]\left(\frac{r^2}{r+1}\right)$ $h(t) = 0.5\cos(2t - 0.5) + \cos(t) + 2$ $r = \sqrt{x_1^2 + x_2^2}$ $t = \text{atan}\,2(x_1, x_2)$		2	[−20,20]	0
20	$f_{\text{Scahffer1}}(x) = .5 + \frac{\sin^2\left(x_1^2+x_2^2\right)^2 - .5}{1+.001\left(x_1^2+x_2^2\right)^2}$	Scahffer1	2	[−100,100]	0

FLA algorithm comes in first rank with HGSO algorithm for Best and Average results, and in second place for Average results after HGSO algorithm and followed by HGS which comes in third place.

3.1.5. Results for fixed dimensions multi-modal problems

The detailed results (Best, Worst, Median, Average, and Std) are presented in Tables (S7 & S8) in the supplementary material file. The summary of the results obtained from the proposed FLA algorithm and the rival algorithms is given in Table 10. It is clear from Table 10 that the proposed FLA algorithm is superior to all competing algorithms for both best and average obtained results. In regards to best results, it can be seen that the proposed FLA algorithm obtains better results than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 0, 9, 4, 0, 4, 8, 3, 4, 5 and 9 problems, respectively. The proposed algorithm is worse than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 test problems, respectively. In regards to the average results as it can be seen in Table 10, FLA obtains superior results to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpS in for 0, 9, 4, 0, 4, 9, 4, 4, 6 and 9 problems, respectively. The proposed FLA is inferior to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 0, 0, 0, 0, 0, 0, 0, 0, 0, and 0 test functions, respectively.

The Wilcoxon test is performed to determine the significant difference between two competing algorithms with the results presented in Table 10. It is concluded from Table 10 that, the proposed FLA is significantly better than all the rival algorithms for both best and average results, which is confirmed by the *P*-value column in Table 10. As a further analysis, the Friedman rank test is conducted to rank all the rival algorithms, with the results presented in Table 11. It is clear from Table 11 that the proposed FLA algorithm comes in the first rank with HGS and HGSO algorithms for both Best and Average results followed by HHO which comes in second place for the Best and Average results.

Table 4
Comparison summary between the proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Wilcoxon test for 30D.

Algorithms	Criteria	Better	Equal	Worse	P-value	Dec.
FLA vs. HGS	Best	1	9	1	0.655	≈
	Average	6	5	0	0.043	+
FLA vs. AEO	Best	11	0	0	0.003	+
	Average	11	0	0	0.000	+
FLA vs. HHO	Best	7	4	0	0.018	+
	Average	7	4	0	0.018	+
FLA vs. HGSO	Best	0	10	1	0.317	≈
	Average	0	10	1	0.317	≈
FLA vs. TEO	Best	8	3	0	0.012	+
	Average	8	3	0	0.012	+
FLA vs. SSA	Best	11	0	0	0.000	+
	Average	11	0	0	0.000	+
FLA vs. WOA	Best	9	2	0	0.008	+
	Average	9	2	0	0.008	+
FLA vs. SCA	Best	11	0	0	0.000	+
	Average	11	0	0	0.000	+
FLA vs. GSA	Best	11	0	0	0.003	+
	Average	11	0	0	0.000	+
FLA vs. LshapecnEpSin	Best	11	0	0	0.003	+
	Average	11	0	0	0.000	+

Table 5
The proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Friedman rank test for 30D.

Algorithms	Rank	
	Best	Average
FLA	2.45	2.25
HGS	2.86	3.75
AEO	11.00	11.00
HHO	3.73	3.50
HGSO	2.36	2.25
TEOFile	4.68	4.45
SSA	8.36	8.10
WOA	5.18	5.20
SCA	8.18	8.60
GSA	7.18	7.00
LshapecnEpSin	10	9.90

Table 6
Comparison summary between the proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Wilcoxon test for 50D.

Algorithms	Criteria	Better	Equal	Worse	P-value	Dec.
FLA vs. HGS	Best	1	9	1	0.655	≈
	Average	5	5	1	0.173	≈
FLA vs. AEO	Best	11	0	0	0.003	+
	Average	11	0	0	0.000	+
FLA vs. HHO	Best	7	4	0	0.018	+
	Average	7	4	0	0.018	+
FLA vs. HGSO	Best	0	10	1	0.317	≈
	Average	0	10	1	0.317	≈
FLA vs. TEO	Best	8	3	0	0.012	+
	Average	8	3	0	0.012	+
FLA vs. SSA	Best	11	0	0	0.000	+
	Average	11	0	0	0.000	+
FLA vs. WOA	Best	9	2	0	0.008	+
	Average	9	2	0	0.008	+
FLA vs. SCA	Best	11	0	0	0.000	+
	Average	11	0	0	0.000	+
FLA vs. GSA	Best	11	0	0	0.003	+
	Average	11	0	0	0.000	+
FLA vs. LshapecnEpSin	Best	11	0	0	0.005	+
	Average	11	0	0	0.000	+

Table 7
The proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Friedman rank test for 50D.

Algorithms	Rank	
	Best	Average
FLA	2.40	2.20
HGS	2.95	3.85
AEO	11.00	11.00
HHO	3.60	3.60
HGSO	2.40	2.20
TEOFile	4.55	4.25
SSA	8.00	7.60
WOA	5.30	5.30
SCA	8.70	8.70
GSA	7.10	7.40
LshapecnEpSin	10.00	9.90

Table 8
Comparison summary between the proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Wilcoxon test for 100D.

Algorithms	Criteria	Better	Equal	Worse	P-value	Dec.
FLA vs. HGS	Best	1	9	1	0.655	≈
	Average	6	4	1	0.225	≈
FLA vs. AEO	Best	11	0	0	0.003	+
	Average	11	0	0	0.000	+
FLA vs. HHO	Best	7	4	0	0.018	+
	Average	7	4	0	0.018	+
FLA vs. HGSO	Best	0	10	1	0.317	≈
	Average	0	9	2	0.180	≈
FLA vs. TEO	Best	7	3	1	0.123	≈
	Average	7	3	1	0.161	≈
FLA vs. SSA	Best	11	0	0	0.000	+
	Average	11	0	0	0.000	+
FLA vs. WOA	Best	9	2	0	0.008	+
	Average	9	2	0	0.008	+
FLA vs. SCA	Best	11	0	0	0.000	+
	Average	11	0	0	0.000	+
FLA vs. GSA	Best	11	0	0	0.003	+
	Average	11	0	0	0.000	+
FLA vs. LshapecnEpSin	Best	11	0	0	0.003	+
	Average	11	0	0	0.000	+

Table 9
The proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Friedman rank test for 100D.

Algorithms	Rank	
	Best	Average
FLA	2.40	2.30
HGS	2.75	3.75
AEO	11.00	11.00
HHO	3.60	3.50
HGSO	2.40	2.20
TEOFile	4.55	4.45
SSA	7.40	7.25
WOA	5.30	5.20
SCA	8.50	8.90
GSA	8.10	7.75
LshapecnEpSin	10	9.70

To evaluate the FLA algorithms' stability, convergence comparisons between LSHADE-EpSin, GSA, SCA, WOA, SSA, TEO, HGSO, HHO, AEO, HGS and FLA for optimization of some classical problems are plotted in Fig. 7. The convergence curve relates to the fitness values and the number of iterations for all competing algorithms. This curve shows the greatest fitness value $[f]$ at each iteration $[t]$, and it continues in this manner until the algorithm's maximum number of iterations is reached. Furthermore, one of the most critical graphical investigations for evaluating

Table 10
Comparison summary between the proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Wilcoxon test for Fixed dimensions multi-modal problems.

Algorithms	Criteria	Better	Equal	Worse	P-value	Dec.
FLA vs. HGS	Best	0	9	0	1.000	≈
	Average	0	9	0	1.000	≈
FLA vs. AEO	Best	9	0	0	0.008	+
	Average	9	0	0	0.008	+
FLA vs. HHO	Best	4	5	0	0.068	≈
	Average	4	5	0	0.068	≈
FLA vs. HGSO	Best	0	9	0	1.000	≈
	Average	0	9	0	1.000	≈
FLA vs. TEO	Best	4	5	0	0.068	≈
	Average	4	5	0	0.068	≈
FLA vs. SSA	Best	8	1	0	0.012	+
	Average	9	0	0	0.008	+
FLA vs. WOA	Best	3	6	0	0.109	≈
	Average	4	5	0	0.068	+
FLA vs. SCA	Best	4	5	0	0.068	≈
	Average	4	5	0	0.068	≈
FLA vs. GSA	Best	5	4	0	0.043	+
	Average	6	3	0	0.028	+
FLA vs. LshapecnEpSin	Best	9	0	0	0.003	+
	Average	9	0	0	0.008	+

Table 11
The proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Friedman rank test for Fixed dimensions multi-modal problems.

Algorithms	Rank	
	Best	Average
FLA	3.44	3.28
HGS	3.44	3.28
AEO	10.78	10.89
HHO	4.50	4.56
HGSO	3.44	3.28
TEOFile	4.94	4.56
SSA	8.56	8.78
WOA	4.56	4.83
SCA	5.61	5.44
GSA	6.50	7.22
LshapecnEpSin	10.22	9.89

the performance of competing algorithms is determining the optimization algorithm's capacity to find the solution fast. This investigation aids in the analysis of optimization algorithms' convergence rates. It is therefore very necessary to examine FLA's convergence for the optimum solutions across iteration numbers, as shown in Fig. 5.

Additionally, to an in-depth analysis of the FLA algorithm, the affecting exploration and exploitation capabilities over the classical functions were carried out. Fig. 6 depicts the exploration and exploitation ratios during the search space while solving the utilized functions for the tests conducted for exploration and exploitation measurements of FLA. As can be seen from the plotted curves in Fig. 6, during the majority of the search process, FLA maintains a trade-off balance between exploration and exploitation ratio.

3.2. CEC'17 test suit analysis

In this section, the performance of the proposed FLA is further evaluated and tested using CEC2017 suite [84], which is one of the more difficult benchmarks that have been used in the Congress on Evolutionary Computation conference to test the performance of many algorithms. CEC2017 functions are classified into four classes: unimodal ($F1$ and $F3$), multiple ($F4-F10$), hybrid ($F11-F20$), and lastly, composition functions (F21-F30). The specification of these functions are presented in Table 12. The FLA has been evaluated with these benchmarks and compared with these well-regarded algorithms such as LSHADE-EpSin, GSA, SCA, WOA, SSA, TEO, HGSO, HHO, AEO, and HGS. To have a fair comparison, all algorithms were run independently 30 times with a stopping condition equal to $10\,000 \times D$, where D is the number of decision variables. Also, each algorithm starts with a population of 30 search agents randomly generated.

3.2.1. Results for 10D

The detailed results (Best, Worst, Median, Average, and Std) are presented in Tables (S9) in the supplementary material file. The summary of the results obtained from the proposed FLA algorithm and the rival algorithms is given in Table 13. It is clear from Table 13 that the proposed FLA algorithm is superior to all competing algorithms for both best and average obtained results. In regards to best results, it can be seen that the proposed FLA algorithm obtains better results than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 17, 29, 28, 28, 29, 18, 27, 29, 25 and 29 problems, respectively. The proposed algorithm is worse than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 8, 0, 1, 1, 0, 5, 1, 0, 3, 1 test problems, respectively. In regards to the average results as it can be seen in Table 15, FLA obtains superior results to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpS in for 26, 29, 29, 28, 29, 25, 29, 29, 25 and 29 problems, respectively. The proposed FLA is inferior to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 3, 0, 0, 1, 0, 4, 0, 0, 4, and 0 test functions, respectively.

The Wilcoxon test is performed to determine the significant difference between two competing algorithms with the results presented in Table 13. It is concluded from Table 13 that, the proposed FLA is significantly better than all the rival algorithms for both best and average results, which is confirmed by the P-value column in Table 13. As a further analysis, the Friedman rank test is conducted to rank all the rival algorithms, with the results presented in Table 14. It is clear from Table 14 that the proposed FLA algorithm comes in the first rank for both Best and Average results followed by HGS which comes in second place for the Best and SSA algorithm for the Average results.

3.2.2. Results for 30D

The detailed results (Best, Worst, Median, Average, and Std) are presented in Tables (S10) in the supplementary material file, with the summary of the results presented in Table 15. From that table and considering the best obtained results, the proposed FLA algorithm obtains better results than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin in 22, 29, 29, 29, 29, 22, 28, 29, 28 and 29 test functions respectively. The proposed FLA algorithm obtains worse results than HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpSin for 5, 0, 0, 0, 0, 6, 1, 0, 0, 0 test functions respectively. In regards to the average results as shown in Table 15, FLA obtains superior results to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpS in for 25, 29, 29, 28, 29, 24, 29, 29, 28 and 29 function, respectively. The proposed FLA is inferior to HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADECnEpS in 4, 0, 0, 1, 0, 4, 0, 0, 1, 0 problems, respectively.

For more analysis, the Wilcoxon test is conducted in order to see the significance difference between two algorithms with the results presented in Table 15. It is clear from Table 15 that, the proposed FLA is significantly better than all other competing algorithms which is confirmed by the P-value column in Table 15. As a further analysis, the Friedman rank test is carried out to

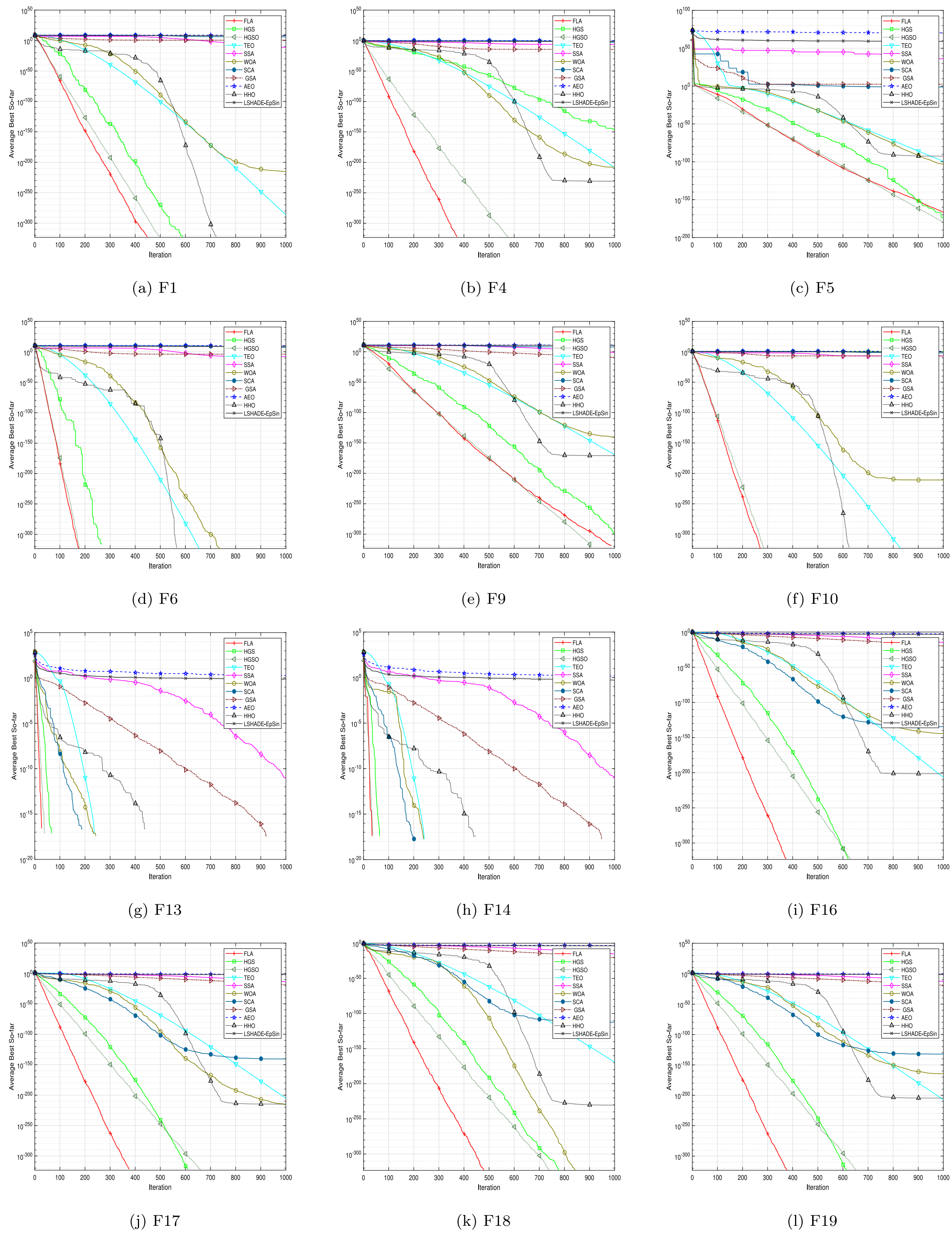


Fig. 5. Convergence behavior of the competitive.

rank all the competing algorithms, with the results presented in Table 16. It is clear from Table 16 that the proposed FLA algorithm is ranked first for both Best and Average results followed by HGS, which comes in second place for both Best and Average results.

To evaluate the FLA algorithms' stability, convergence comparisons between LSHADE-EpSin, GSA, SCA, WOA, SSA, TEO, HGSO, HHO, AEO, HGS and FLA for optimization of CEC'17 test suite are plotted in Fig. 7. To avoid the issue of local optima, composite functions were utilized to evaluate the balance between exploration and exploitation. It is clear from Fig. 7 that the proposed FLA algorithms performed better than the rival algorithms and reached a better solution faster than them.

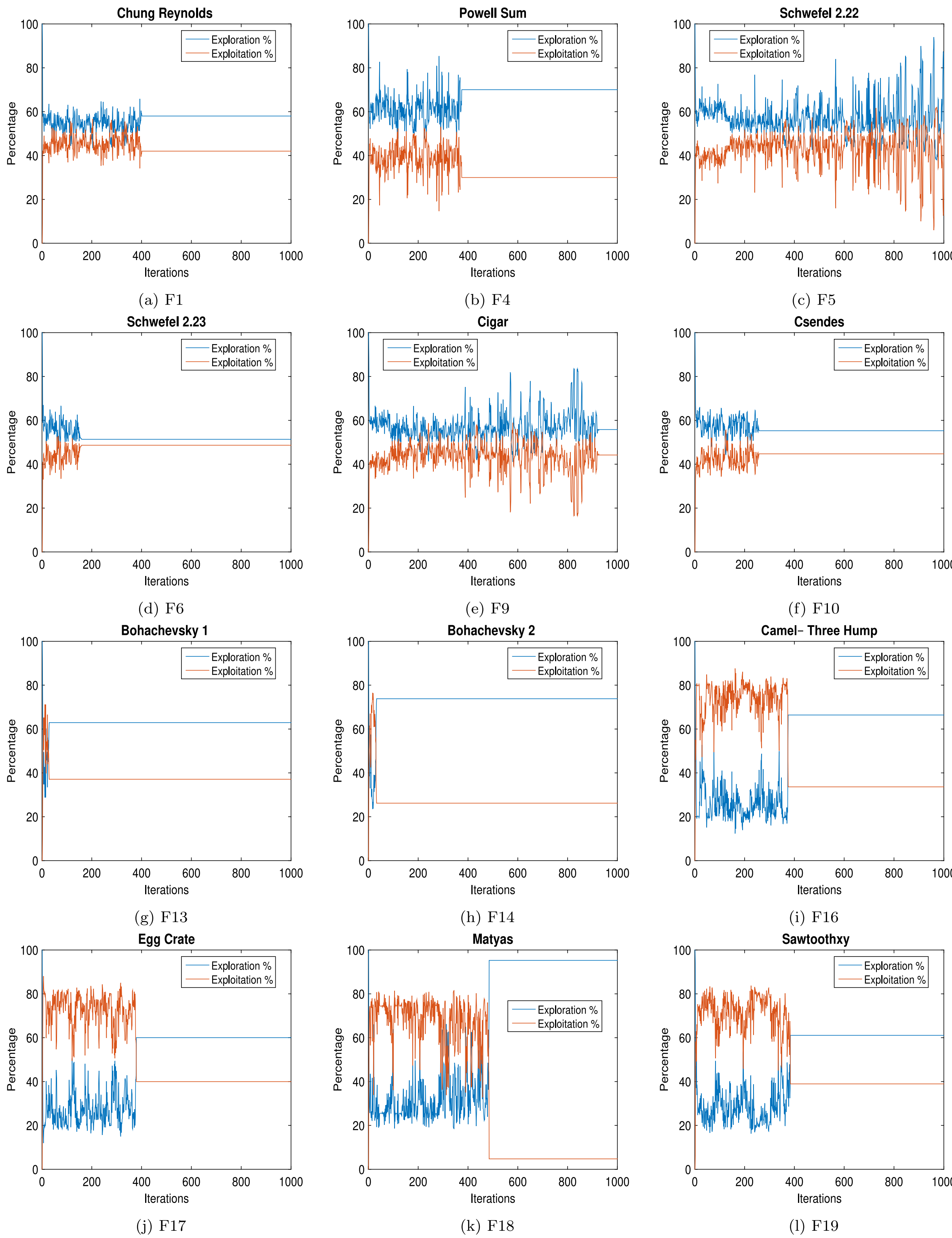


Fig. 6. Exploration and exploitation phases for FLA in some benchmark optimization functions.

Additionally, to an in-depth analysis of the FLA algorithm, the affecting exploration and exploitation capabilities over the CEC2017 test suite were carried out. Fig. 7 depicts the exploration and exploitation ratios during the search space while solving the utilized functions for the tests conducted for exploration and exploitation measurements of FLA. As can be seen from the plotted curves in Fig. 8, during the majority of the search process, FLA maintains a trade-off balance between exploration and exploitation ratio.

3.3. *Comparing FLA with recent developed metaheuristics*

In this subsection, We compare Our developed algorithm with other 3 powerful and recently introduced algorithms. These algorithms are

Table 12
Review of CEC2017 benchmark function problems.

Type	No.	Description	Fi
Unimodal functions	1	Shifted and rotated bent cigar function	100
	3	Shifted and rotated Zakharov function	300
Simple multimodal functions	4	Shifted and rotated Rosenbrock's function	400
	5	5 Shifted and rotated Rastrigin's function	500
	6	Shifted and rotated expanded Scaffer's F6 function	600
	7	Shifted and rotated Lunacek Bi-Rastrigin function	700
	8	Shifted and rotated Non-continuous Rastrigin's function	800
	9	Shifted and rotated Levy function	900
	10	Shifted and rotated Schwefel's function	1000
Hybrid functions	11	Hybrid function 1 (N = 3	1100
	12	Hybrid function 2 (N = 3)	1200
	13	Hybrid function 3 (N = 3)	1300
	14	Hybrid function 4 (N = 4)	1400
	15	Hybrid function 5 (N = 4)	1500
	16	Hybrid function 6 (N = 4)	1600
	17	Hybrid function 6 (N = 5)	1700
	18	Hybrid function 6 (N = 5)	1800
	19	Hybrid function 6 (N = 5)	1900
	20	Hybrid function 6 (N = 6)	2000
Composition functions	21	Composition function 1 (N = 3)	2100
	22	Composition function 2 (N = 3)	2200
	23	Composition function 3 (N = 4)	2300
	24	Composition function 4 (N = 4)	2400
	25	Composition function 5 (N = 5)	2500
	26	Composition function 6 (N = 5)	2600
	27	Composition function 7 (N = 6)	2700
	28	Composition function 8 (N = 6)	2800
	29	Composition function 9 (N = 3)	2900
	30	Composition function 10 (N = 3)	3000

Table 13
Comparison summary between the proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Wilcoxon test for 10D.

Algorithms	Criteria	Better	Equal	Worse	*P*-value	Dec.
FLA vs. HGS	Best	17	4	8	0.003	+
	Average	26	0	3	0.000	+
FLA vs. AEO	Best	29	0	0	0.000	+
	Average	29	0	0	0.000	+
FLA vs. HHO	Best	28	0	1	0.000	+
	Average	29	0	0	0.000	+
FLA vs. HGSO	Best	28	0	1	0.000	+
	Average	28	0	1	0.000	+
FLA vs. TEO	Best	29	0	0	0.000	+
	Average	29	0	0	0.000	+
FLA vs. SSA	Best	18	6	5	0.000	+
	Average	25	1	3	0.000	+
FLA vs. WOA	Best	27	1	1	0.000	+
	Average	29	0	0	0.000	+
FLA vs. SCA	Best	29	0	0	0.000	+
	Average	29	0	0	0.000	+
FLA vs. GSA	Best	25	1	3	0.000	+
	Average	25	0	4	0.000	+
FLA vs. LshapecnEpSin	Best	28	0	1	0.000	+
	Average	29	0	0	0.000	+

- Farmland fertility [85]
- African Vultures Optimization Algorithm [86]
- Gorilla Troops Optimizer [87]

The parameters of these algorithms are also given in Table 2. We use 29 functions from IEEE CEC 2017. The results are included in table (S11). From this table it can be noticed that FLA has a good results comparing to the 3 other optimizers. FLA achieves the minimum values in seven different values namely ($F5, F7, F8, F10, F21, F23, F29$) and the second best in ten different functions namely ($F3, F6, F9, F16, F17, F20, F24,$

Table 14
The proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Friedman rank test for 10D.

Algorithms	Rank	
	Best	Average
FLA	1.90	1.40
HGS	2.45	3.24
AEO	9.66	9.17
HHO	6.33	7.83
HGSO	6.93	5.78
TEOFile	9.90	10.55
SSA	2.81	2.91
WOA	4.33	5.98
SCA	6.05	4.97
GSA	7.21	6.55
LshapecnEpSin	8.45	7.62

$F25, F26, F27$). Overall by applying Friedman testy we can noticed that FLA results are better than FF, AVOA, and GTO.

3.4. Exploitative and exploratory assessments of FLA

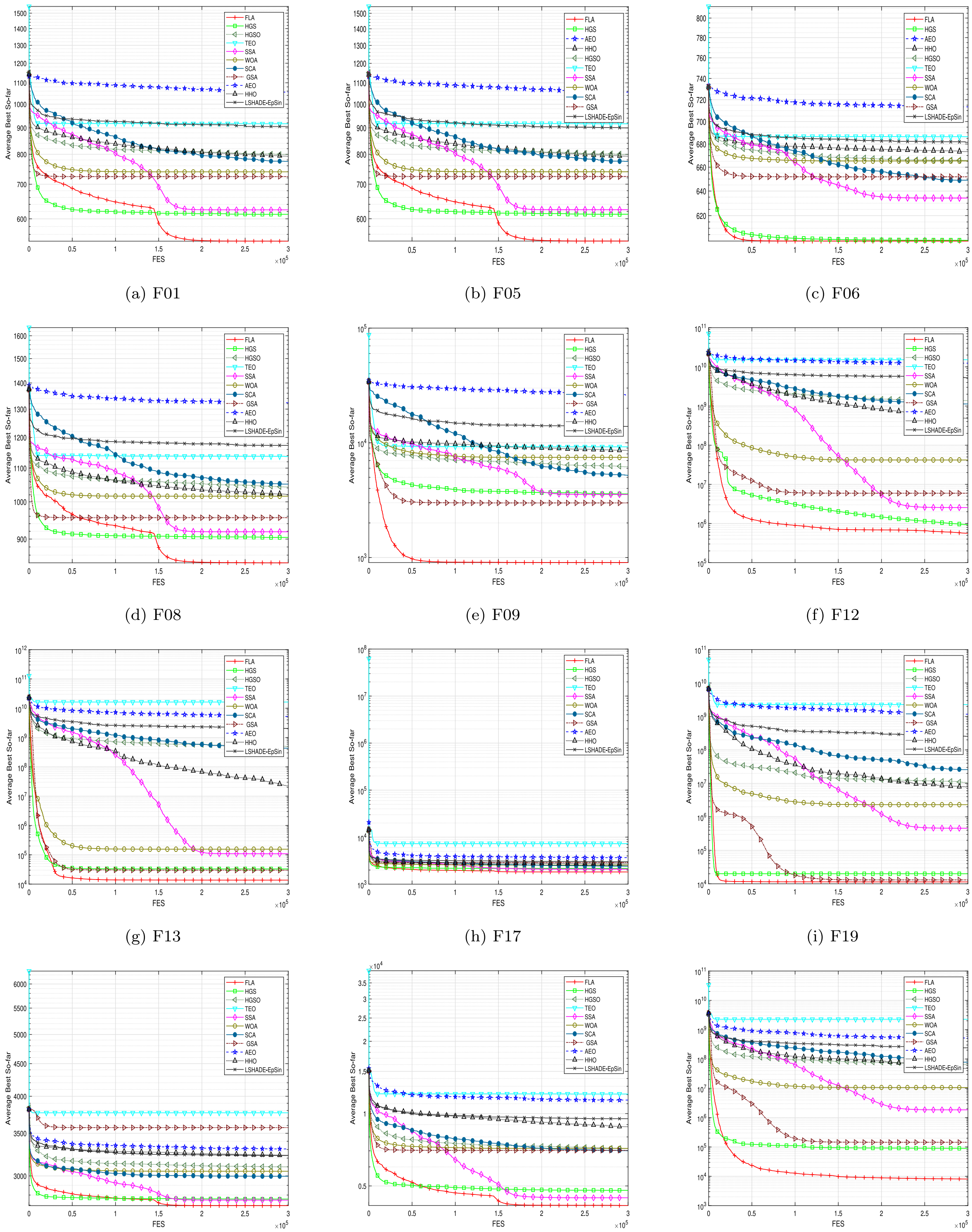
In this subsection, an FLA assessment is done to test exploitative and exploratory abilities for all used benchmark.

3.4.1. Exploitative behavior assessment

Testing FLA using many unimodal functions ($F1-F7$ in Table 3 and $F1\&F3$ in Table 12) can help in test FLA ability in exploitation since they have only one global solution. By analyzing the results of classical functions ($F1 - F7$) and the CEC 2017 benchmark functions ($F1\&F3$). Using many dimensions in both benchmark, we can see FLA has a strong exploitation abilities.

3.4.2. Exploratory behavior assessment

Testing FLA using many multimodal functions ($F8 - F11$ in Table 3 and $F4 - F10$) in Table 12 can help in test FLA ability in exploration since they have many sub-optimal solutions. By



(a) F01 (b) F05 (c) F06

(d) F08 (e) F09 (f) F12

(g) F13 (h) F17 (i) F19

(j) F22 (k) F26 (l) F30

Fig. 7. Convergence behavior of the competitive algorithms on CEC2017 benchmark functions.

analyzing the results of classical functions ($F1 - F7$) and the CEC 2017 benchmark functions ($F1\&F3$). Using many dimensions in both benchmark, we can see FLA has a strong exploration abilities.

3.5. Engineering optimization problems

In general, optimization issue in the engineering design is a prominent study area, and literature uses many optimization methods to tackle this kind of problem [88–92]. This section evaluates the FLA algorithm's performance in real-world engineering applications such as tension/compression spring design, welded beam construction, pressure vessel design, speed reducer design, and three-Bar truss design. These problems have been addressed using FLA and comparing results against those of other competing algorithms. The FLA and competing algorithms were run 30 separate times with 1000 total iterations to arrive at

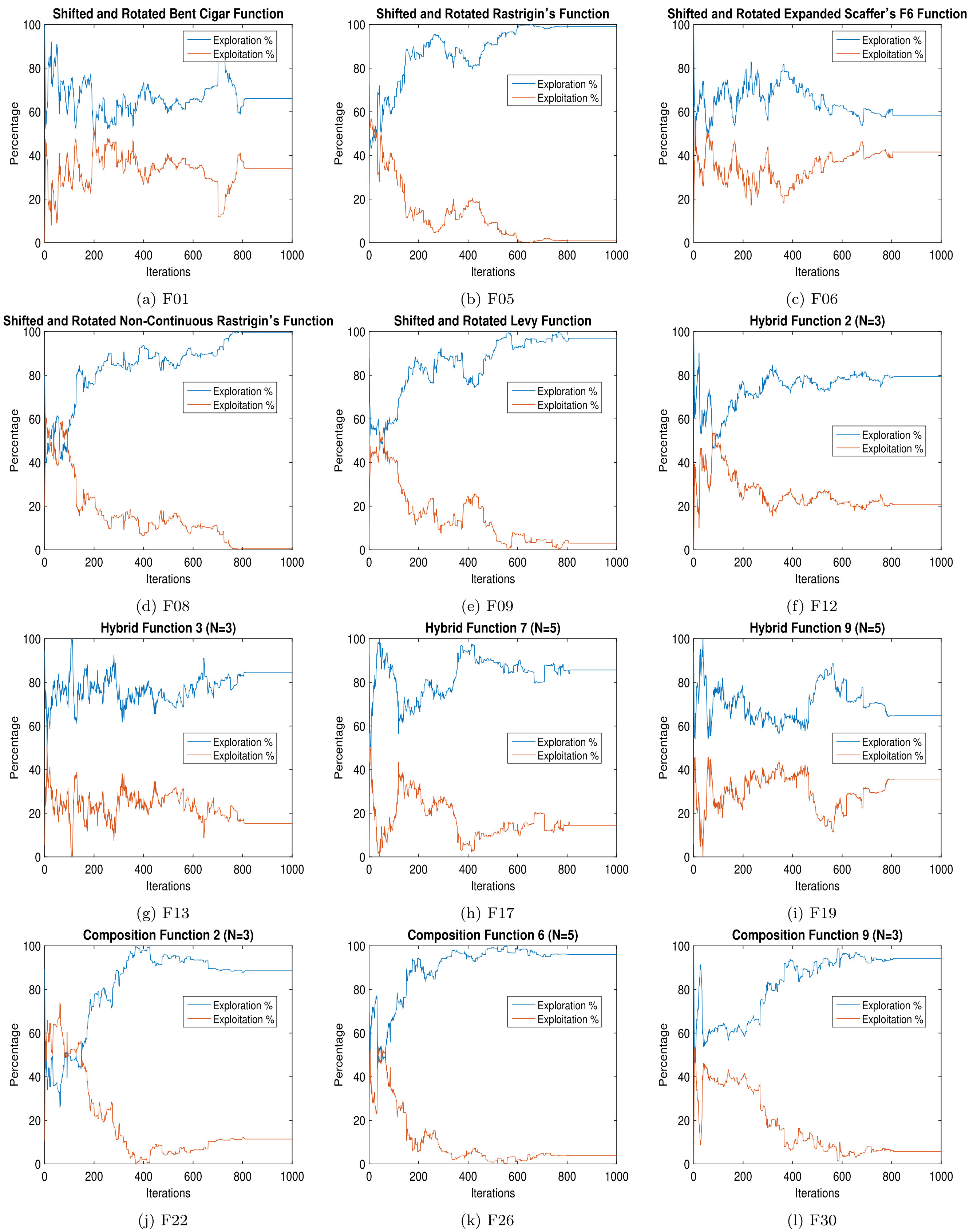


Fig. 8. Exploration and exploitation phases of the competitive algorithms on CEC2017 benchmark functions.

Table 15
Comparison summary between the proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Wilcoxon test for 30D.

Algorithms	Criteria	Better	Equal	Worse	*P*-value	Dec.
FLA vs. HGS	Best	22	2	5	0.005	+
	Average	25	0	4	0.000	+
FLA vs. AEO	Best	29	0	0	0.000	+
	Average	29	0	0	0.000	+
FLA vs. HHO	Best	29	0	0	0.000	+
	Average	29	0	0	0.000	+
FLA vs. HGSO	Best	29	0	0	0.000	+
	Average	28	0	1	0.000	+
FLA vs. TEO	Best	29	0	0	0.000	+
	Average	29	0	0	0.000	+
FLA vs. SSA	Best	22	1	6	0.001	+
	Average	24	1	4	0.001	+
FLA vs. WOA	Best	28	0	1	0.000	+
	Average	29	0	0	0.000	+
FLA vs. SCA	Best	29	0	0	0.000	+
	Average	29	0	0	0.000	+
FLA vs. GSA	Best	28	1	0	0.000	+
	Average	28	0	1	0.000	+
FLA vs. LshapecnEpSin	Best	29	0	0	0.000	+
	Average	29	0	0	0.000	+

Table 16
Comparison summary between the proposed FLA, HGS, AEO, HHO, HGSO, TEO, SSA, WOA, SCA, GSA, and LSHADEcnEpSin based on Friedman rank test for 30D.

Algorithms	Rank	
	Best	Average
FLA	1.48	1.36
HGS	2.28	2.62
AEO	10.24	10.21
HHO	6.84	7.29
HGSO	6.52	6.21
TEOFile	10.24	10.41
SSA	2.64	2.67
WOA	4.84	5.69
SCA	6.55	6.07
GSA	5.26	4.69
LshapecnEpSin	9.10	8.78

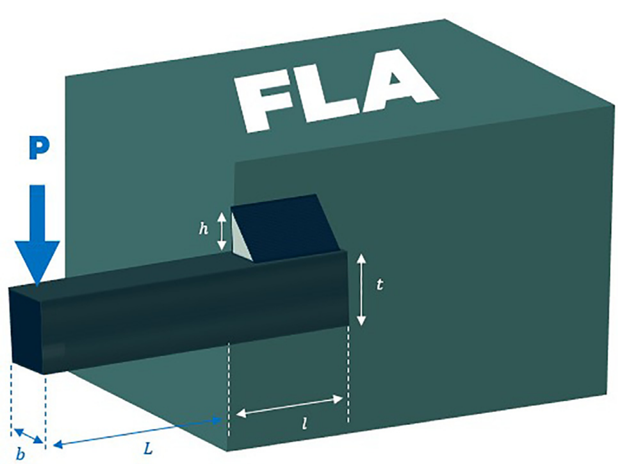


Fig. 9. Welded beam design problem.

a fair comparison. The 5 problems used have more inequality constraints. Here, we use a plenty function to reject any solution violates any of the problem pre-define constraints.

3.5.1. *Welded beam design problem*

The main goal of this problem is to identify the minimum manufacturing cost of welded beam design. The design of this problem is shown in Fig. 9 that depends mainly on selecting the optimal values of different design variables, which are four optimization variables, namely, the length of the attached part of the bar (l), the thickness of the weld (h), the height of the bar (t), and the thickness of the bar (b). The mathematical form is given in Eq. (39).

$$
\begin{array}{ll}
\text{Consider} & \vec{x} = [x_1 \quad x_2 \quad x_3 \quad x_4] = [h \quad l \quad tt \quad bb] \\
\text{Minimize} & f(\vec{x}) = 1.10471 x_1^2 x_2 \\
& \quad +0.04811 x_3 x_4 \,(14.0 + x_2) \\
\text{Subject to} & g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \leqslant 0 \\
& g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \leqslant 0 \\
& g_3(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \leqslant 0 \\
& g_4(\vec{x}) = x_1 - x_4 \leqslant 0 \\
& g_5(\vec{x}) = P - P_c(\vec{x}) \leqslant 0 \\
& g_6(\vec{x}) = 0.125 - x_1 \leqslant 0 \\
& g_7(\vec{x}) = 1.10471 x_1^2 \\
& \quad +0.04811 x_3 x_4 \,(14.0 + x_2) - 5.0 \leqslant 0 \\
\text{Variables range} & 0.1 \leqslant x_1 \leqslant 2 \\
& 0.1 \leqslant x_2 \leqslant 10 \\
& 0.1 \leqslant x_3 \leqslant 10 \\
& 0.1 \leqslant x_4 \leqslant 2 \\
\text{where} & \tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau'' \frac{x_2}{2R} + (\tau'')^2} \\
& \tau' = \frac{p}{\sqrt{2x_1x_2}}, \quad \tau'' = \frac{MR}{J} \\
& M = P\left(L + \frac{x_2}{2}\right) \\
& R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2} \\
& J = 2\left\{\sqrt{2x_1x_2}\left[\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2\right]\right\} \\
& \sigma(\vec{x}) = \frac{6PL}{x_4 x_3^2}, \quad \delta(\vec{x}) = \frac{6PL^3}{E x_3^2 x_4} \\
& P_c(\vec{x}) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right) \\
& P = 6000 \text{ lb}, L = 14 \text{ in.}, \quad \delta_{\max} = 0.25 \text{ in.} \\
& E = 30 \times 1^6 \text{ psi}, \quad G = 12 \times 10^6 \text{ psi} \\
& \tau_{\max} = 13\,600 \text{ psi}, \quad \sigma_{\max} = 30\,000 \text{ psi}
\end{array} \tag{34}
$$

The proposed FLA algorithm is used to solve the welded beam design problem, and its performance is compared to various literature-published optimization techniques. Table 17 illustrates a comparison of the top competitive algorithm solutions. Furthermore, the statistical findings are compared in Table 18. It can be concluded from these results that the FLA performed better than all other algorithms and had the lowest total cost.

3.5.2. *Tension/compression spring design problem*

The tension/compression spring design problem has the primary goal of finding the lowest weight of the tension/compression spring that satisfies its design requirements, including shear stress, surge frequency, and deflection. The three design variables to be taken into consideration, as illustrated in Fig. 10, are wire diameter (d), mean coil diameter (D), and the number of active coils (N). The mathematical form is presented in eq. (39)

Table 17
Results obtained from competitive algorithms for the welded beam problem.

Algorithm	Optimal values for variables				Optimal cost
	H	l	t	B	
LSHADE-EpSin	0.2884	3.1057	9.3491	0.2999	2.17E+00
GSA	0.2194	3.4462	8.3911	0.2386	1.91E+00
SCA	0.1947	3.7831	9.1234	0.2077	1.78E+00
WOA	0.329	2.5471	6.8078	0.3789	2.37E+00
SSA	0.707	1.5736	4.66	0.7736	2.20E+00
TEO	0.2606	4.9671	5.0117	0.6744	2.26E+00
HGSO	0.2402	3.8964	7.9686	0.2679	2.09E+00
HHO	0.2134	3.5601	8.4629	0.2346	1.77E+00
AEO	0.7764	0.3832	40.3196	200	5.88E+03
HGS	0.8397	0.4142	43.595	158.9344	5.99E+03
FLA	0.1983	3.6664	9.0705	0.2057	**1.75E+00**

Table 18
The results obtained from competitive algorithms for the welded beam problem.

Algorithm	Best	Mean	Worst	Std Dev
LSHADE-EpSin	2.17E+00	2.416526	2.11E+01	4.49E+01
GSA	1.91E+00	2.31E+00	2.91E+00	2.37E−01
SCA	1.78E+00	1.94E+00	2.07E+00	7.40E−02
WOA	2.37E+00	2.67E+00	3.32E+00	2.14E−01
SSA	2.20E+00	3.71E+00	5.88E+00	1.06E+00
TEO	2.26E+00	4.87E+00	3.26E+05	7.54E+04
HGSO	2.09E+00	2.32E+00	2.61E+00	1.86E−01
HHO	1.77E+00	1.88E+00	2.23E+00	1.25E−01
AEO	5.88E+03	5.90E+03	6.10E+03	4.74E+01
HGS	5.99E+03	6.67E+03	7.30E+03	5.43E+02
FLA	**1.75E+00**	**1.82E+00**	**1.93E+00**	**5.29E−02**

Table 19
Results obtained from competitive algorithms for the tension/ compression spring problem.

Algorithm	d	D	N	*Optimal cost*
LSHADE-EpSin	0.0752	0.9971	3.266	2.97E−02
GSA	0.05	0.317	14.0971	1.28E−02
SCA	0.0524	0.3727	10.4533	1.27E−02
WOA	0.0523	0.3723	10.4317	1.27E−02
SSA	0.0526	0.3786	10.1265	1.27E−02
TEO	0.0548	0.4358	7.855	1.29E−02
HGSO	0.0557	0.4493	7.7172	1.35E−02
HHO	0.0529	0.3873	9.6975	1.27E−02
AEO	0.0517	0.3577	11.2326	1.27E−02
HGS	0.05	0.3174	14.0306	1.27E−02
FLA	0.0499	0.315	14.3045	1.27E−02

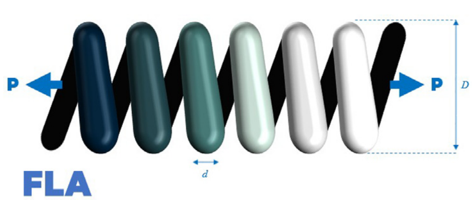


Fig. 10. Tension/compression spring design problem.

Table 20
The results obtained from competitive algorithms for the tension/compression spring problem.

Algorithm	Best	Mean	Worst	Std Dev
LSHADE-EpSin	2.97E−02	1.79E+02	2.34E+06	6.18E+05
GSA	1.28E−02	1.01E+01	2.52E+02	4.60E+01
SCA	1.27E−02	**1.32E−02**	1.38E−02	2.81E−04
WOA	1.27E−02	1.39E−02	1.64E−02	1.09E−03
SSA	1.27E−02	1.36E+05	1.24E+06	2.71E+05
TEO	1.29E−02	2.26E+05	1.58E+06	4.16E+05
HGSO	1.35E−02	1.47E−02	1.68E−02	7.30E−04
HHO	1.27E−02	1.37E−02	1.58E−02	8.88E−04
AEO	1.27E−02	**1.32E−02**	1.34E−02	2.29E−04
HGS	1.27E−02	1.01E+04	1.76E+05	3.83E+04
FLA	1.27E−02	**1.32E−02**	1.48E−02	4.75E−04

$$
\begin{aligned}
&\text{Consider} && \vec{x} = [x_1 x_2 x_3] = [dDN],\\
&\text{Minimize} && f(\vec{x}) = (x_3 + 2)\, x_2 x_1^2,\\
&\text{Subject to} && g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leqslant 0,\\
& && g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} \leqslant 0,\\
& && g_3(\vec{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leqslant 0,\\
& && g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leqslant 0,\\
&\text{Variables range} && 0.05 \leqslant x_1 \leqslant 2\\
& && 0.25 \leqslant x_2 \leqslant 1.30\\
& && 2.00 \leqslant x_3 \leqslant 15
\end{aligned}
\tag{35}
$$

This engineering issue (tension/compressed spring design) is solved using the proposed FLA, and a comparison of the best competitive algorithm solutions is presented in Table 19. The statistical findings obtained are also shown in Table 19. As revealed from Table 20, the optimum value for function is 0.012665 achieved using the FLA algorithm; FLA has outperformed all other competing algorithms.

3.5.3. Pressure vessel design problem

The main goal of this problem is to identify the minimum cost of raw materials of pressure vessel design problem. This problem depends mainly on selecting the optimal values of different design variables, which are divided into four optimization variables: inner radius R, head of thickness T_h, cylindrical section length vessel L, and shell thickness T_s. Fig. 11 describe it design and the mathematical form is given as follows:

$$
\begin{aligned}
&\text{Consider} && \vec{x} = [x_1 x_2 x_3 x_4],\\
&\text{Minimize} && f(\vec{x}) = 0.6224 x_1 x_3 x_4 + 1.7781 x_2 x_3^2\\
& && \quad +3.1661 x_1^2 x_4 + 19.84 x_1^2 x_3,\\
&\text{Subject to} && g_1(x) = -x_1 + 0.0193 x_3 \leqslant 0,\\
& && g_2(x) = -x_2 + 0.00954 x_3 \leq 0,\\
& && g_3(x) = -\pi x_3^2 x_4 - \tfrac{4}{3}\pi x_3^3 + 1,296,000 \leq 0,\\
& && g_4(x) = x_4 - 240 \leq 0,\\
&\text{Variables range} && 0 \leq x_1, x_2 \leq 100 \text{ and } 10 \leq x_3, x_4 \leq 200
\end{aligned}
\tag{36}
$$

The proposed FLA algorithm is used to solve the pressure vessel design problem, and its performance is compared to various literature-published optimization techniques. Table 21 illustrates a comparison of the top competitive algorithm solutions. Furthermore, the statistical findings are compared in Table 22. It can be

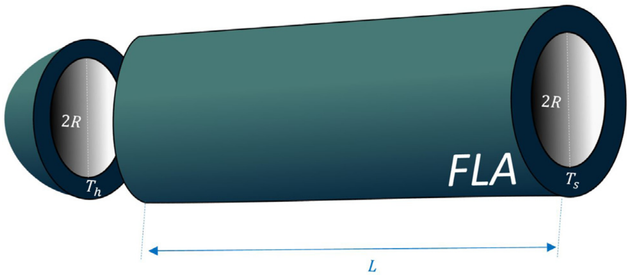


Fig. 11. Pressure vessel design problem.

Table 21
Results obtained from competitive algorithms for the Pressure vessel design.

	Best	x1	x2	x3	x4
LSHADE-EpSin	6061.0777	0.8125	0.4375	42.091266	176.7465
GSA	6020.651	0.774696	0.334423	40.8342	196.7131
SCA	5870.124	0.774549	0.383204	40.31962	194.3434
WOA	5940.184	0.774547	0.38321	40.31962	199.9999
SSA	6879.703	0.859051	0.473962	43.62445	180.7057
TEO	5892.124	0.773249	0.383204	40.31962	200
HGSO	5870.124	0.778149	0.383204	40.31962	192.78
HHO	5876.941	0.772346	0.385079	40.52191	197.203
AEO	5882.124	0.7723449	0.383404	40.31782	194.54
HGS	5921.124	0.773339	0.383204	40.31924	195.87
FLA	5871.124	0.772129	0.393504	40.31962	200

Table 22
The results obtained from competitive algorithms for the Pressure vessel design.

Algorithms	min	max	mean	STD
LSHADE-EpSin	6061.0777	6664.0237	6244.7657	1.24E−12
GSA	6020.651	6368.888	6034.022	128.428
SCA	5870.124	6666.339	6145.893	219.693
WOA	5940.184	6154.355	6070.147	0.047518
SSA	66 879.703	8758.232	6942.32	342.971
TEO	5892.124	5927.798	5873.266	12.27243
HGSO	5870.124	7301.196	6370.616	482.3923
HHO	5876.941	6898.222	6242.955	248.7349
AEO	5882.124	7104.345	6423.236	455.1753
HGS	5921.124	6865.196	6345.333	349.2151
FLA	5871.124	7301.196	6295.391	439.8161

Table 23
The results obtained from competitive algorithms for the speed beam design.

Algorithms	min	max	mean	STD
LSHADE-EpSin	3050.634	3693.634258	3293.634258	4.54747E
GSA	2995.635	2997.654658	2996.639946	0.05604814
SCA	3047.561	3081.626661	3064.509498	2.925883629
WOA	3030.563	3075.609846	3044.283808	21.39236888
SSA	3071.526	3072.333892	3072.306978	0.147413346
TEO	2997.9157	2998.634258	2998.204258	4.62521E−13
HGSO	3067.561	3069.634258	3068.634258	0.68603
HHO	2997.8157	2997.964258	2997.834258	2.67037E−13
AEO	3029.002	3062.626661	3044.476422	4.783277262
HGS	3019.5833	3073.626661	3051.500418	13.67234678
FLA	2993.634	3002.626661	2996.500418	5.296278268

concluded from these results that the FLA performed better than all other algorithms and had the lowest total cost.

3.5.4. Speed reducer design problem

The main goal of this problem is to identify the minimum reducer weight of speed reducer design. This problem depends mainly on selecting the optimal values of different design variables, namely, face width b, teeth module m, teeth number in the pinion z, first shaft length between bearings l_1, second shaft length between bearings l_2 and diameter of the first and second shafts d_1, d_2. Fig. 12 describe it design and the mathematical form is given as follows:

$$
\begin{aligned}
&\text{Consider} && \vec{x} = [z_1 z_2 z_3 z_4 z_5 z_6 z_7],\\
&\text{Minimize} && f(\vec{z}) = 0.7854 z_1 z_2^2 \left(3.3333 z_3^2 + 14.9334 z_3 \right.\\
& && \left. -43.0934\right) - 1.508 z_1\\
& && \left(z_6^2 + z_7^2\right) + 7.4777\left(z_6^3 + z_7^3\right)\\
& && +0.7854\left(z_4 z_6^2 + z_5 z_7^2\right)\\
&\text{Subject to} && g_1(\vec{z}) = \tfrac{27}{z_1 z_2^2 z_3} - 1 \le 0\\
& && g_2(\vec{z}) = \tfrac{397.5}{z_1 z_2^2 z_3} - 1 \le 0\\
& && g_3(\vec{z}) = \tfrac{1.93 z_4^3}{z_2 z_3 z_6^4} - 1\\
& && g_4(\vec{z}) = \tfrac{1.93 z_5^3}{z_2 z_3 z_7^4} - 1 \le 0\\
& && g_5(\vec{z}) = \tfrac{1}{110 z_6^3}\sqrt{\left(\tfrac{745 z_4}{z_2 z_3}\right)^2 + 16.9 \times 10^6} - 1\\
& && \le 0\\
& && g_6(\vec{z}) = \tfrac{1}{85 z_7^3}\sqrt{\left(\tfrac{745 z_5}{z_2 z_3}\right)^2 + 157.5 \times 10^6} - 1\\
& && \le 0\\
& && g_7(\vec{z}) = \tfrac{z_2 z_3}{40} - 1 \le 0\\
& && g_8(\vec{z}) = \tfrac{5 z_2}{z_1} - 1 \le 0\\
& && g_9(\vec{z}) = \tfrac{z_1}{12 z_2} - 1 \le 0\\
& && g_{10}(\vec{z}) = \tfrac{1.5 z_6 + 1.9}{z_4} - 1 \le 0\\
& && g_{11}(\vec{z}) = \tfrac{1.1 z_7 + 1.9}{z_5} - 1 \le 0\\
&\text{Variables range} && 2.6 \le z_1 \le 3.6,\ 0.7 \le z_2 \le 0.8,\\
& && 17 \le z_3 \le 28,\ 7.3 \le z_4 \le 8.3,\\
& && 7.8 \le z_5 \le 8.3,\ 2.9 \le z_6 \le 3.9,\\
& && and\ 5 \le z_7 \le 5.5
\end{aligned}
\tag{37}
$$

The proposed FLA algorithm is used to solve the speed reducer design problem, and its performance is compared to various literature-published optimization techniques. Table 23 illustrates a comparison of the top competitive algorithm solutions. Furthermore, the statistical findings are compared in Table 24. It can be concluded from these results that the FLA performed better than all other algorithms and had the lowest total cost.

3.5.5. Three-bar truss design problem

The main goal of this problem is to identify the minimum reducer weight of three-bar truss design. This problem depends mainly on selecting the optimal values of two variables. Fig. 13 describe it design and the mathematical form is given as follows:

Minimize $f(\overrightarrow{x}) = (2\sqrt{2}x_1 + x_2) * l$

Subject to:

$$g_1(\overrightarrow{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0$$

$$g_2(\overrightarrow{x}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0$$

$$g_3(\overrightarrow{x}) = \frac{1}{\sqrt{2}x_2 + 2x_1}P - \sigma \le 0$$

where $l = 100$ cm, $P = 2$ kN/cm^2, $\sigma = 2$ kN/cm^2, $0 \le x_1, x_2 \le 1$

Table 24
The results obtained from competitive algorithms for the speed beam design.

	Best	x1	x2	x3	x4	x5	x6	x7
LSHADE-EpSin	3050.634	3.597488	0.7	17	7.322	7.713554	3.352356	5.274721
GSA	2995.635	3.497632	0.7	17	7.3	7.713441	3.35006	5.285641
SCA	3047.561	3.52516	0.7	17	8.352	7.8	3.6220	5.3465
WOA	3030.563	3.50875	0.7	17	7.3	7.8	3.46102	5.28921
SSA	3071.526	3.6	0.7	17	7.3	8.072148	3.402879	5.312241
TEO	2997.9157	3.50384	0.7	17	7.3	7.72933	3.35649	5.2867
HGSO	3067.561	3.51025	0.7	17	8.35	7.8	3.36220	5.28772
HHO	2997.9157	3.50384	0.7	17	7.3	7.72933	3.35649	5.2867
AEO	3029.002	3.52012	0.7	17	8.37	7.8	3.36697	5.28871
HGS	3019.58336	3.5	0.7	17	7.3	7.67039	3.54242	5.24581
FLA	2993.634	3.497599	0.7	17	7.3	7.713535	3.350056	5.285631

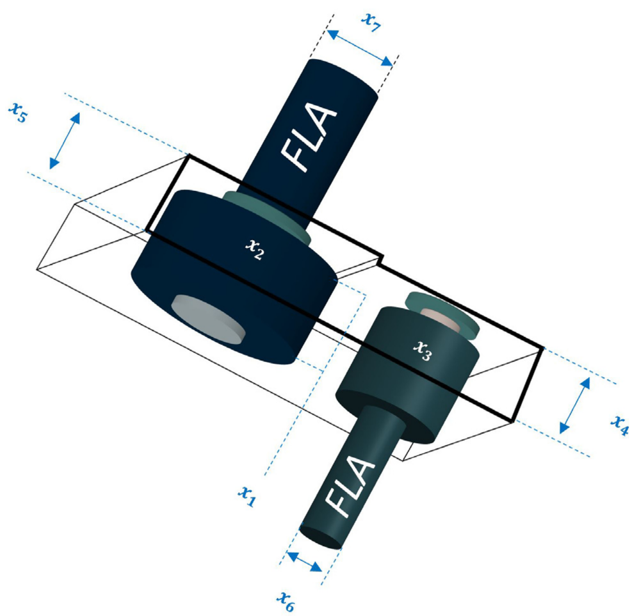


Fig. 12. Speed reducer design problem.

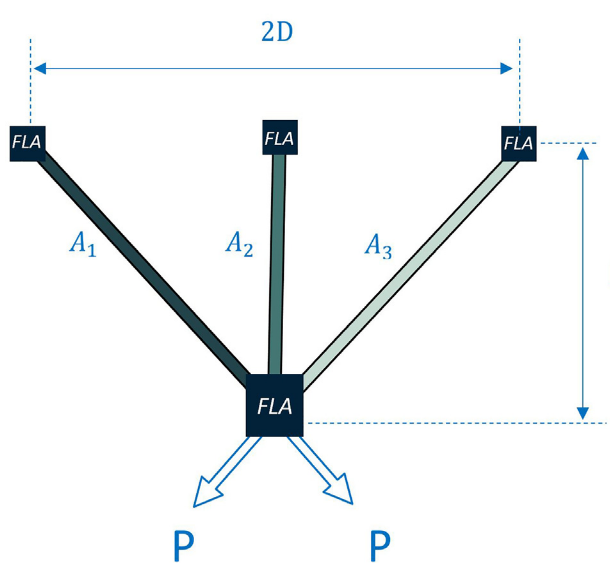


Fig. 13. Three-bar design problem.

The proposed FLA algorithm is used to solve the speed reducer design problem, and its performance is compared to various literature-published optimization techniques. Table 25 illustrates a comparison of the top competitive algorithm solutions. Furthermore, the statistical findings are compared in Table 26. It can be concluded from these results that the FLA performed better than all other algorithms and had the lowest total cost.

Table 25
The results obtained from competitive algorithms for the Three-bar truss design problem.

	Best	x1	x2
LSHADE-EpSin	263.8915	0.785249	0.410335
GSA	263.8916	0.789005	0.407224
SCA	263.8715	0.788649	0.408235
WOA	263.8958	0.78860276	0.408453070000000
SSA.	264.8067	0.756483	0.508411
TEO	263.8958	0.7886618	0.4082831
HGSO	264.1762	0.778254	0.440528
HHO	263.8958	0.78866541	0.408275784
AEO	263.9154	0.79369	0.39426
HGS	263.8959	0.7884562	0.40886831
FLA	263.8552	0.78645	0.41369

Table 26
The results obtained from competitive algorithms for the Three-bar truss design problem.

Algorithms	min	max	mean	STD
LSHADE-EpSin	263.8915	263.8915	263.8915	1.25E−13
GSA	263.8916	263.909	263.8949	0.004166
SCA	263.8715	263.8915	263.8915	2.59E−4
WOA	263.8958	263.8958	263.8958	1.14E−13
SSA	264.8067	265.9765	264.879	0.067512
TEO	263.8958	263.8958	263.8958	1.17E−13
HGSO	264.1762	283.1862	272.1714	5.778049
HHO	263.8958	263.8958	263.8958	3.61E−07
AEO	263.9154	263.9154	263.9154	1.73E−12
HGS	263.8959	263.8959	263.8959	1.69E−12
FLA	263.8552	263.8552	263.8552	1.89E−12

4. Conclusion and potential future researches

In general, the various critical considerations related to the design and performance of the optimization method are simplicity, robustness, and flexibility. An optimization algorithm may be widely accepted in the scientific community if it adequately addresses these characteristics. In this paper, a new physics-based metaheuristic algorithm, named Fick's optimization algorithm (FLA), has been developed. This algorithm is based on the concept of Fick's laws of diffusion. In the FLA, the optimization procedures are represented in 3 states, namely the diffusion state, equilibrium state, and steady state. Each of them has been represented in the suggested method. A series of various optimization problems were utilized to verify the developed FLA's capacity to identify the optimum solution. These problems include 20 classical benchmark functions, 29 functions from the CEC2017 benchmark, and two engineering problems. It has been noticed from the benchmark functions' statistical findings that the FLA yielded better results than other well-known metaheuristic algorithms. Additionally, because of the empirical study of engineering problems, it can ascertain the FLA's applicability in solving difficulties encountered in real-world applications. Although FLA has demonstrated its superiority and effectiveness in solving many problems with various dimensions, it still, like all metaheuristic algorithms, may have a slow convergence in some problems, especially complex and high-dimensional ones.

This prior debate that examined the proposed FLA's superiority may have a significant impact on opening a wide range of future works. FLA may be used for a wide range of applications, including image processing applications, PV parameter estimation, feature selection, text and data mining applications, big data applications, smart home applications, resource management applications, industry and engineering applications, task scheduling, and others. For real-world applications, FLA may apply to the optimization of discrete, binary, and multiple goals. In addition, improving FLA's performance may include using the principles of levy flight, disruption, mutation, or other stochastic components.

CRediT authorship contribution statement

Fatma A. Hashim: Propose the algorithm and its mathematical equation. **Reham R. Mostafa:** Writing – editing, Doing the experiments. **Abdelazim G. Hussien:** Review the paper, Investigation, Validation, Handling comments. **Seyedali Mirjalili:** Supervision, Giving comments. **Karam M. Sallam:** Writing – editing, Doing the experiments.

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.

Data availability

Data will be made available on request.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.knosys.2022.110146.

References

[1] Vijay Kumar, Jitender Kumar Chhabra, Dinesh Kumar, Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems, J. Comput. Sci. 5 (2) (2014) 144–155.

[2] Farhad Soleimanian Gharehchopogh, An improved tunicate swarm algorithm with best-random mutation strategy for global optimization problems, J. Bionic Eng. (2022) 1–26.

[3] Simrandeep Singh, Harbinder Singh, Nitin Mittal, Abdelazim G Hussien, Filip Sroubek, A feature level image fusion for night-vision context enhancement using arithmetic optimization algorithm based image segmentation, Expert Syst. Appl. (2022) 118272.

[4] Farhad Soleimanian Gharehchopogh, Advances in tree seed algorithm: A comprehensive survey, Arch. Comput. Methods Eng. (2022) 1–24.

[5] Song-Quan Ong, Gomesh Nair, Ragheed Duraid Al Dabbagh, Nur Farihah Aminuddin, Putra Sumari, Laith Abualigah, Heming Jia, Shubham Mahajan, Abdelazim G Hussien, Diaa Salama Abd Elminaam, Comparison of pre-trained and convolutional neural networks for classification of jackfruit artocarpus integer and artocarpus heterophyllus, in: Classification Applications with Deep Learning and Machine Learning Technologies, Springer, 2023, pp. 129–141.

[6] Jinhao Zhang, Mi Xiao, Liang Gao, Quanke Pan, Queuing search algorithm: A novel metaheuristic algorithm for solving engineering optimization problems, Appl. Math. Model. 63 (2018) 464–490.

[7] Wenhu Zhao, Chengbin Du, Shouyan 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.

[8] Shafih Ghafori, Farhad Soleimanian Gharehchopogh, Advances in spotted hyena optimizer: a comprehensive survey, Arch. Comput. Methods Eng. (2021) 1–22.

[9] Hamid Reza Rafat Zaman, Farhad Soleimanian Gharehchopogh, An improved particle swarm optimization with backtracking search optimization algorithm for solving continuous optimization problems, Eng. Comput. (2021) 1–35.

[10] Abdelazim G Hussien, Aboul Ella Hassanien, Essam H. Houssein, Mohamed Amin, Ahmad Taher Azar, New binary whale optimization algorithm for discrete optimization problems, Eng. Optim. 52 (6) (2020) 945–959.

[11] Abualigah Laith, Multi-verse optimizer algorithm: a comprehensive survey of its results, variants, and applications, Neural Comput. Appl. 32 (16) (2020) 12381–12401.

[12] Shuang Wang, Abdelazim G. Hussien, Heming Jia, Laith Abualigah, Rong Zheng, Enhanced remora optimization algorithm for solving constrained engineering optimization problems, Mathematics 10 (10) (2022) 1696.

[13] Abdelazim G. Hussien, Laith Abualigah, Raed Abu Zitar, Fatma A. Hashim, Mohamed Amin, Abeer Saber, Khaled H. Almotairi, Amir H. Gandomi, Recent advances in harris hawks optimization: A comparative study and applications, Electronics 11 (12) (2022) 1919.

[14] Vinícius Veloso de Melo, Wolfgang Banzhaf, Drone Squadron Optimization: a novel self-adaptive algorithm for global numerical optimization, Neural Comput. Appl. 30 (10) (2018) 3117–3144.

[15] Rong Zheng, Abdelazim G. Hussien, He-Ming Jia, Laith Abualigah, Shuang Wang, Di Wu, An improved wild horse optimizer for solving optimization problems, Mathematics 10 (8) (2022) 1311.

[16] Abdelazim G. Hussien, Mohamed Amin, Mohamed Abd El Aziz, A comprehensive review of moth-flame optimisation: variants, hybrids, and applications, J. Exp. Theor. Artif. Intell. 32 (4) (2020) 705–725.

[17] Laith Abualigah, Amir H. Gandomi, Mohamed Abd Elaziz, Abdelazim G. Hussien, Ahmad M. Khasawneh, Mohammad Alshinwan, Essam H. Houssein, Nature-inspired optimization algorithms for text document clustering—a comprehensive analysis, Algorithms 13 (12) (2020) 345.

[18] Abdelazim G. Hussien, Mohamed Amin, A self-adaptive Harris Hawks optimization algorithm with opposition-based learning and chaotic local search strategy for global optimization and feature selection, Int. J. Mach. Learn. Cybern. 13 (2) (2022) 309–336.

[19] Mina Javanmard Goldanloo, Farhad Soleimanian Gharehchopogh, A hybrid OBL-based firefly algorithm with symbiotic organisms search algorithm for solving continuous optimization problems, J. Supercomput. 78 (3) (2022) 3998–4031.

[20] Laith Abualigah, Group search optimizer: a nature-inspired meta-heuristic optimization algorithm with its results, variants, and applications, Neural Comput. Appl. 33 (7) (2021) 2949–2972.

[21] Abdelazim G. Hussien, An enhanced opposition-based salp swarm algorithm for global optimization and engineering problems, J. Ambient Intell. Humaniz. Comput. 13 (1) (2022) 129–150.

[22] Sukanta Nama, Apu Kumar Saha, Sima Ghosh, Parameters optimization of geotechnical problem using different optimization algorithm, Geotech. Geol. Eng. 33 (5) (2015) 1235–1253.

[23] Afshin Faramarzi, Mohammad Heidarinejad, Brent Stephens, Seyedali Mirjalili, Equilibrium optimizer: A novel optimization algorithm, Knowl.-Based Syst. 191 (2020) 105190.

[24] Mehdi Abedi, Farhad Soleimanian Gharehchopogh, An improved opposition based learning firefly algorithm with dragonfly algorithm for solving continuous optimization problems, Intell. Data Anal. 24 (2) (2020) 309–338.

[25] Hanaa Fathi, Hussain AlSalman, Abdu Gumaei, Ibrahim I.M. Manhrawy, Abdelazim G. Hussien, Passent El-Kafrawy, An efficient cancer classification model using microarray and high-dimensional data, Comput. Intell. Neurosci. 2021 (2021).

[26] Adel Saad Assiri, Abdelazim G. Hussien, Mohamed Amin, Ant lion optimization: variants, hybrids, and applications, IEEE Access 8 (2020) 77746–77764.

[27] John Henry Holland, et al., Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence, MIT Press, 1992.

[28] John R. Koza, Genetic Programming II: Automatic Discovery of Reusable Programs, MIT Press, 1994.

[29] Rainer Storn, Differential Evolution-A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces, Vol. 11, Technical Report, International Computer Science Institute, 1995.

[30] Hans-Georg Beyer, Hans-Paul Schwefel, Evolution strategies–a comprehensive introduction, Nat. Comput. 1 (1) (2002) 3–52.

[31] Amir Hossein Gandomi, Amir Hossein Alavi, Krill herd: a new bio-inspired optimization algorithm, Commun. Nonlinear Sci. Numer. Simul. 17 (12) (2012) 4831–4845.

[32] Laith Abualigah, Mohammad Shehab, Mohammad Alshinwan, Seyedali Mirjalili, Mohamed Abd Elaziz, Ant lion optimizer: a comprehensive survey of its variants and applications, Arch. Comput. Methods Eng. 28 (3) (2021) 1397–1416.
[33] Xin-She Yang, Amir Hossein Gandomi, Bat algorithm: a novel approach for global engineering optimization, Eng. Comput. (2012).
[34] M. Khishe, Mohammad Reza Mosavi, Chimp optimization algorithm, Expert Syst. Appl. 149 (2020) 113338.
[35] Shimin Li, Huiling Chen, Mingjing Wang, Ali Asghar Heidari, Seyedali Mirjalili, Slime mould algorithm: A new method for stochastic optimization, Future Gener. Comput. Syst. 111 (2020) 300–323.
[36] Seyedali Mirjalili, Amir H. Gandomi, Seyedeh Zahra Mirjalili, Shahrzad Saremi, Hossam Faris, Seyed Mohammad Mirjalili, Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems, Adv. Eng. Softw. 114 (2017) 163–191.
[37] Ali Asghar Heidari, Seyedali Mirjalili, Hossam Faris, Ibrahim Aljarah, Majdi Mafarja, Huiling Chen, Harris hawks optimization: Algorithm and applications, Future Gener. Comput. Syst. 97 (2019) 849–872.
[38] Ali Kaveh, Neda Farhoudi, A new optimization method: Dolphin echolocation, Adv. Eng. Softw. 59 (2013) 53–70.
[39] Reham R. Mostafa, Abdelazim G. Hussien, Muhammad Attique Khan, Seifedine Kadry, Fatma A. Hashim, Enhanced coot optimization algorithm for dimensionality reduction, in: 2022 Fifth International Conference of Women in Data Science at Prince Sultan University (WiDS PSU), IEEE, 2022, pp. 43–48.
[40] Dawid Połap, Marcin Woźniak, Red fox optimization algorithm, Expert Syst. Appl. 166 (2021) 114107.
[41] Fatma A. Hashim, Abdelazim G. Hussien, Snake Optimizer: A novel meta-heuristic optimization algorithm, Knowl.-Based Syst. 242 (2022) 108320.
[42] Abdelazim G. Hussien, Ali Asghar Heidari, Xiaojia Ye, Guoxi Liang, Huiling Chen, Zhifang Pan, Boosting whale optimization with evolution strategy and Gaussian random walks: an image segmentation method, Eng. Comput. (2022) 1–45.
[43] Vahideh Hayyolalam, Ali Asghar Pourhaji Kazem, Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems, Eng. Appl. Artif. Intell. 87 (2020) 103249.
[44] Afshin Faramarzi, Mohammad Heidarinejad, Seyedali Mirjalili, Amir H. Gandomi, Marine Predators Algorithm: A nature-inspired metaheuristic, Expert Syst. Appl. 152 (2020) 113377.
[45] Sushmita Sharma, Apu Kumar Saha, Arindam Majumder, Sukanta Nama, MPBOA-A novel hybrid butterfly optimization algorithm with symbiosis organisms search for global optimization and image segmentation, Multimedia Tools Appl. 80 (8) (2021) 12035–12076.
[46] Sanjoy Chakraborty, Sukanta Nama, Apu Kumar Saha, An improved symbiotic organisms search algorithm for higher dimensional optimization problems, Knowl.-Based Syst. 236 (2022) 107779.
[47] Huangjing Yu, Heming Jia, Jianping Zhou, Abdelazim G. Hussien, Enhanced Aquila optimizer algorithm for global optimization and constrained engineering problems, Math. Biosci. Eng. 19 (12) (2022) 14173–14211.
[48] Laith Abualigah, Mohamed Abd Elaziz, Putra Sumari, Zong Woo Geem, Amir H. Gandomi, Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer, Expert Syst. Appl. 191 (2022) 116158.
[49] Scott Kirkpatrick, C. Daniel Gelatt, Mario P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680.
[50] Berat Doğan, Tamer Ölmez, A new metaheuristic for numerical function optimization: Vortex Search algorithm, Inform. Sci. 293 (2015) 125–145.
[51] Esmat Rashedi, Hossein Nezamabadi-Pour, Saeid Saryazdi, GSA: a gravitational search algorithm, Inform. Sci. 179 (13) (2009) 2232–2248.
[52] Seyedali Mirjalili, Seyed Mohammad Mirjalili, Abdolreza Hatamlou, Multi-verse optimizer: a nature-inspired algorithm for global optimization, Neural Comput. Appl. 27 (2) (2016) 495–513.
[53] Fatma A. Hashim, Essam H. Houssein, Mai S. Mabrouk, Walid Al-Atabany, Seyedali Mirjalili, Henry gas solubility optimization: A novel physics-based algorithm, Future Gener. Comput. Syst. 101 (2019) 646–667.
[54] Laith Abualigah, Mohamed Abd Elaziz, Abdelazim G. Hussien, Bisan Alsalibi, Seyed Mohammad Jafar Jalali, Amir H. Gandomi, Lightning search algorithm: a comprehensive survey, Appl. Intell. 51 (4) (2021) 2353–2376.
[55] Abdelazim G. Hussien, Fatma A. Hashim, Raneem Qaddoura, Laith Abualigah, Adrian Pop, An enhanced evaporation rate water-cycle algorithm for global optimization, Processes 10 (11) (2022) 2254.
[56] Mohamed Abdel-Basset, Reda Mohamed, Karam M. Sallam, Ripon K. Chakrabortty, Light spectrum optimizer: A novel physics-inspired metaheuristic optimization algorithm, Mathematics 10 (19) (2022) 3466.
[57] Iman Ahmadianfar, Omid Bozorg-Haddad, Xuefeng Chu, Gradient-based optimizer: A new metaheuristic optimization algorithm, Inform. Sci. 540 (2020) 131–159.
[58] Abdelazim G. Hussien, Mohamed Amin, Mingjing Wang, Guoxi Liang, Ahmed Alsanad, Abdu Gumaei, Huiling Chen, Crow search algorithm: theory, recent advances, and applications, IEEE Access 8 (2020) 173548–173565.
[59] Esmaeil Atashpaz-Gargari, Caro Lucas, Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition, in: 2007 IEEE Congress on Evolutionary Computation, Ieee, 2007, pp. 4661–4667.
[60] R. Venkata Rao, Vimal J. Savsani, D.P. Vakharia, Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems, Comput. Aided Des. 43 (3) (2011) 303–315.
[61] Abdelazim G. Hussien, Diego Oliva, Essam H. Houssein, Angel A. Juan, Xu Yu, Binary whale optimization algorithm for dimensionality reduction, Mathematics 8 (10) (2020) 1821.
[62] Abdelazim G. Hussien, Aboul Ella Hassanien, Essam H. Houssein, Siddhartha Bhattacharyya, Mohamed Amin, S-shaped binary whale optimization algorithm for feature selection, in: Recent Trends in Signal and Image Processing, Springer, 2019, pp. 79–87.
[63] Ahmed G. Gad, Karam M. Sallam, Ripon K. Chakrabortty, Michael J. Ryan, Amr A. Abohany, An improved binary sparrow search algorithm for feature selection in data classification, Neural Comput. Appl. (2022) 1–49.
[64] Shubhra Dwivedi, Manu Vardhan, Sarsij Tripathi, Building an efficient intrusion detection system using grasshopper optimization algorithm for anomaly detection, Cluster Comput. 24 (3) (2021) 1881–1900.
[65] Nebojsa Bacanin, Timea Bezdan, K. Venkatachalam, Fadi Al-Turjman, Optimized convolutional neural network by firefly algorithm for magnetic resonance image classification of glioma brain tumor grade, J. Real-Time Image Process. 18 (4) (2021) 1085–1098.
[66] Laith Abualigah, Ali Diabat, A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments, Cluster Comput. 24 (1) (2021) 205–223.
[67] Ismail M. Ali, Karam M. Sallam, Nour Moustafa, Ripon Chakraborty, Michael J. Ryan, Kim-Kwang Raymond Choo, An automated task scheduling model using non-dominated sorting genetic algorithm II for fog-cloud systems, IEEE Trans. Cloud Comput. (2020).
[68] Essam H. Houssein, Bahaa El-Din Helmy, Ahmed A. Elngar, Diaa Salama Abdelminaam, Hassan Shaban, An improved tunicate swarm algorithm for global optimization and image segmentation, IEEE Access 9 (2021) 56066–56092.
[69] Sanjoy Chakraborty, Apu Kumar Saha, Sukanta Nama, Sudhan Debnath, COVID-19 X-ray image segmentation by modified whale optimization algorithm with population reduction, Comput. Biol. Med. 139 (2021) 104984.
[70] Mahmoud G. Hemeida, Abdalla Ahmed Ibrahim, Al-Attar A. Mohamed, Salem Alkhalaf, Ayman M. Bahaa El-Dine, Optimal allocation of distributed generators DG based Manta Ray Foraging Optimization algorithm (MRFO), Ain Shams Eng. J. 12 (1) (2021) 609–619.
[71] Md. Alamgir Hossain, Karam M. Sallam, Seham S. Elsayed, Ripon K. Chakrabortty, Michael J. Ryan, Optimal power flow considering intermittent solar and wind generation using multi-operator differential evolution algorithm, 2021.
[72] Mohamed Abdel-Basset, Reda Mohamed, Ripon K. Chakrabortty, Karam Sallam, Michael J. Ryan, An efficient teaching-learning-based optimization algorithm for parameters identification of photovoltaic models: Analysis and validations, Energy Convers. Manage. 227 (2021) 113614.
[73] Karam M. Sallam, Md. Alamgir Hossain, Ripon K. Chakrabortty, Michael J. Ryan, An improved gaining-sharing knowledge algorithm for parameter extraction of photovoltaic models, Energy Convers. Manage. 237 (2021) 114030.
[74] Rana Muhammad Adnan, Reham R. Mostafa, Ahmed Elbeltagi, Zaher Mundher Yaseen, Shamsuddin Shahid, Ozgur Kisi, Development of new machine learning model for streamflow prediction: case studies in Pakistan, Stoch. Environ. Res. Risk Assess. (2021) 1–35.
[75] Sanjoy Chakraborty, Sushmita Sharma, Apu Kumar Saha, Ashim Saha, A novel improved whale optimization algorithm to solve numerical optimization and real-world applications, Artif. Intell. Rev. (2022) 1–112.
[76] David H. Wolpert, William G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82.
[77] Noor H. Awad, Mostafa Z. Ali, Ponnuthurai N. Suganthan, Robert G. Reynolds, An ensemble sinusoidal parameter adaptation incorporated with L-SHADE for solving CEC2014 benchmark problems, in: 2016 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2016, pp. 2958–2965.
[78] Seyedali Mirjalili, SCA: a sine cosine algorithm for solving optimization problems, Knowl.-Based Syst. 96 (2016) 120–133.
[79] Seyedali Mirjalili, Andrew Lewis, The whale optimization algorithm, Adv. Eng. Softw. 95 (2016) 51–67.
[80] Abualigah Laith, Shehab Mohammad, Alshinwan Mohammad, Alabool Hamzeh, Salp swarm algorithm: a comprehensive survey, Neural Comput. Appl. 32 (15) (2020) 11195–11215.
[81] A. Kaveh, Armin Dadras, A novel meta-heuristic optimization algorithm: thermal exchange optimization, Adv. Eng. Softw. 110 (2017) 69–84.
[82] Weiguo Zhao, Liying Wang, Zhenxing Zhang, Artificial ecosystem-based optimization: a novel nature-inspired meta-heuristic algorithm, Neural Comput. Appl. 32 (13) (2020) 9383–9425.

[83] Yutao Yang, Huiling Chen, Ali Asghar Heidari, Amir H Gandomi, Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts, Expert Syst. Appl. 177 (2021) 114864.

[84] Guohua Wu, Rammohan Mallipeddi, Ponnuthurai Nagaratnam Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained Real-Parameter Optimization, Technical Report, National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, 2017.

[85] Human Shayanfar, Farhad Soleimanian Gharehchopogh, Farmland fertility: A new metaheuristic algorithm for solving continuous optimization problems, Appl. Soft Comput. 71 (2018) 728–746.

[86] Benyamin Abdollahzadeh, Farhad Soleimanian Gharehchopogh, Seyedali Mirjalili, African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems, Comput. Ind. Eng. 158 (2021) 107408.

[87] Benyamin Abdollahzadeh, Farhad Soleimanian Gharehchopogh, Seyedali Mirjalili, Artificial gorilla troops optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems, Int. J. Intell. Syst. 36 (10) (2021) 5887–5958.

[88] Abdelazim G. Hussien, Essam H. Houssein, Aboul Ella Hassanien, A binary whale optimization algorithm with hyperbolic tangent fitness function for feature selection, in: 2017 Eighth International Conference on Intelligent Computing and Information Systems, ICICIS, IEEE, 2017, pp. 166–172.

[89] Aboul Ella Hassanien, Moataz Kilany, Essam H. Houssein, Hameed AlQaheri, Intelligent human emotion recognition based on elephant herding optimization tuned support vector regression, Biomed. Signal Process. Control 45 (2018) 182–191.

[90] Ahmed A. Ewees, Mohamed Abd Elaziz, Essam H. Houssein, Improved grasshopper optimization algorithm using opposition-based learning, Expert Syst. Appl. 112 (2018) 156–172.

[91] Abdelazim G. Hussien, Aboul Ella Hassanien, Essam H. Houssein, Swarming behaviour of salps algorithm for predicting chemical compound activities, in: 2017 Eighth International Conference on Intelligent Computing and Information Systems, ICICIS, IEEE, 2017, pp. 315–320.

[92] Sushmita Sharma, Apu Kumar Saha, Geetanjali Lohar, Optimization of weight and cost of cantilever retaining wall by a hybrid metaheuristic algorithm, Eng. Comput. 38 (4) (2022) 2897–2923.