ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

An Exponential Map-based Whale Optimization Algorithm (Exp-WOA) for Optimization

Vaibhav Godbole[1] and Shilpa Gaikwad[2]

ABSTRACT

This research work offers three variations of the Whale Optimization Algorithm (WOA) based on exponential chaotic maps, namely Logistic-Exponential-Logistic WOA (LEL-WOA), Logistic-Exponential-Sinusoidal WOA (LES-WOA), and Logistic-Exponential-Tent WOA (LET-WOA). The WOA with an exponential chaos-based mechanism is developed in this study to overcome the poor rate of convergence of the WOA and to prevent getting caught in local optimal solutions while dealing with the challenges. An exponential chaotic mechanism was employed in this research to initialize the agents and control the parameters of the exploration and exploitation phases of WOA. The proposed methodologies (Exp-WOA) are evaluated using twenty-three widely recognized test functions. The results demonstrate that the given solutions can enhance the performance of WOA by achieving optimal (minimum) values. The findings also indicate that LEL-WOA and LES-WOA exhibit faster convergence than WOA.

Article information:

Keywords: Variants of Whale Optimization Algorithm, Exponential Chaotic Maps, Optimization Problem Solution, Swarm Optimization Algorithm, Metaheuristic Algorithm

Article history:
Received: April 15, 2024
Revised: July 11, 2024
Accepted: August 22, 2024
Published: September 14, 2024 (Online)

DOI: 10.37936/ecti-cit.2024184.256409

1. INTRODUCTION

In numerous optimization challenges, it is essential to efficiently determine the best possible solution to a given problem within a reasonable amount of time, even when faced with intricate constraints [1]. Modern intelligent algorithms are frequently used to tackle these optimization challenges [2]. Various other approaches have been suggested to tackle these challenges; nonetheless, they are insufficient in producing improved results. Metaheuristic optimization algorithms have garnered significant interest in scientific communities over the past couple of decades due to their notable advancements in effectively addressing numerous intricate optimization problems. Before metaheuristic optimization algorithms, traditional algorithms such as brute force algorithms [3], greedy algorithms [4], and network flow algorithms [5] etc. were used to solve optimization problems. The conventional methods begin with a single point and require gradient information, therefore resulting in a time-consuming process to attain the global optima. These algorithms proved inadequate in addressing real-world scenarios, such as the localization problem, due to their limited relevance and the complexity of the constraints they imposed. Metaheuristic optimization algorithms mimic biological or physical processes to solve challenging real-world optimization problems [6]. Metaheuristic optimization algorithms have advantages over traditional methods, as mentioned below [7]:

- Traditional methods often use local search techniques that can get stuck in local optima and miss the global best solution. Metaheuristic optimization methods, on the other hand, are designed to look through the whole search space, which makes it less likely that they will get stuck in local optima and more likely that they will find the global optimum.
- Traditional methods usually need to be formulated and changed based on the type of problem, which means they can't be used for all kinds of problems. On the other hand, metaheuristic optimization methods are very flexible and can be used for all kinds of problems without having to make big changes to the core algorithms.
- Traditional methods often make claims about how simple or linear things are that don't hold true in the real world. On the other hand, metaheuristic optimization methods may be able to handle complex, non-linear, and multimodal problems well.

[1,2]The authors are with the Bharati Vidyapeeth (Deemed to be University) College of Engineering, Pune, India, E-mail: professorvaibhav@gmail.com and spgaikwad@bvucoep.edu.in
[1]Corresponding author: professorvaibhav@gmail.com

- Traditional methods are frequently deterministic and may not adequately explore the search space, potentially overlooking better answers; in contrast, metaheuristic optimization methods use stochastic and heuristic approaches to explore the search space, which can lead to new and successful solutions.
- Traditional approaches often focus on single objective optimization or necessitate sophisticated formulations to handle multiple objectives, whereas metaheuristic optimization methods may readily accommodate and optimize many competing objectives at the same time.
- Traditional approaches are frequently less robust, requiring precise data and conditions to perform successfully. In contrast metaheuristic optimization methods are more resilient to uncertainties and alterations in the issue data or environment.

In general, metaheuristic optimization algorithms have several advantages and disadvantages, and the choice of algorithm depends on the characteristics of the optimization problem and the available computing resources [6]. Metaheuristic optimization algorithms are classified into four main categories [6]: i) bio-inspired algorithms [8], ii)mathematics-inspired algorithms [9], iii) physics-inspired algorithms [10], and iv) social-inspired algorithms [11]. Physical algorithms are based on observation and experimentation. Some of the most recognized and widely used physics-inspired optimization algorithms are i) the gravitational search algorithm [12], ii) the black hole optimizer [13], iii) the supernova optimizer [14], iv)doppler effect mean Euclidean distance threshold (DE-MEDT) algorithm [15]. v) the vortex search algorithm [16] and vi) the arithmetic optimization algorithm [17].

Bio-inspired algorithms are inspired by the biological processes and behaviors of living beings in nature [11]. Genetic algorithms, influenced by Darwi's evolution theory, are the most traditional examples of their kind [18]. Some of the most recognized and widely used bio-inspired algorithms are: i) particle swarm optimization (PSO) [19], ii) ant colony optimization (ACO) [20], iii) artificial bee colony (ABC) [21], iv) whale optimization algorithm [22]. The primary source of bio-inspired metaheuristic optimization algorithms is natural interactions between and among species, which might take the form of cooperation or competition. Therefore, most bio-inspired algorithms fall within the category of swarm intelligence algorithms [23].

Above all, because the optimization process is stochastic, the most challenging aspect of developing any meta-heuristic algorithm is striking the right balance between exploration and exploitation [24]. The exploration phase supports the optimizer's effort to thoroughly and globally study the search space [25]. During this phase, the population also experiences some sudden shifts. In contrast, the exploitation phase involves perfecting the feasible answers uncovered during the discovery phase. The population endures slight, rapid variations in this area [25].

The whale optimization algorithm (WOA) is a recently developed nature-inspired meta-heuristic algorithm that imitates the social behavior of humpback whales [26]. A mathematical model of their hunting habits serves as the foundation for the WOA. More specifically, humpback whales use a technique for exploration known as the bubble-net feeding method. The goal of humpback whales is to seek small fish or krill close to the surface. By forming recognizable bubbles along a line that forms a circle or a "9," this foraging is carried out. Whales use two strategies to complete this task: upward spirals and double loops [27]. Three fundamental processes comprise the principal operation of a WOA: encircling the prey, searching for the prey, and using the bubble-net assault approach [26]. Research has demonstrated that this algorithm can outperform other meta-heuristic algorithms in several real-world scenarios, including feature selection, sizing optimization for skeletal structures, cluster head selection, optimal power flow (OPF) in electrical generation systems, and task scheduling. However the basic WOA exhibits several limitations, such as a sluggish convergence rate, a limited precision of solutions, insufficient population variety, and a propensity to settle for the local best solution [1]. To tackle these problems, researchers have proposed several variants of the WOA.

This research aims to introduce an exponential map-based whale optimization algorithm, known as Exp-WOA, for optimization purposes. In this research, the exponential map is introduced to i) generate the initial population and ii) update the exploration and exploitation stages of the original WOA. The optimal positions for search agents are then identified. The proposed Exp-WOA has been evaluated using twenty-three widely recognized benchmark functions.

The rest of the paper is arranged as follows. Section 2 provides an overview of WOA. Section 3 describes the exponential chaotic map. In Section 4, the proposed Exp-WOA has been provided. Section 5 presents the experimental outcomes. Finally, Section 6 presents the conclusion and future work.

2. THE WHALE OPTIMIZATION ALGORITHM

The whale optimization algorithm (WOA) is a stochastic optimization approach emulating the hunting behavior of humpback whales and was created by Mirjalili *et al.* in 2016 [26]. Humpback whales belong to the baleen whale species and are known for their high level of intelligence and ability to experience emotions. The most distinctive attribute of these whales is that they are the largest animal in the world, with a mature length of $39-53$ feet and a weight of $25-30$ metric tons. Spindle cells, characterized by their elongated spindle-shaped bodies, are present in the brain and play a crucial role in facilitating the gregarious, intelligent, and sophisticated behavior observed in whales. The humpback whales have a most diverse hunting habit, and the primary source of their diet is tiny fish. Whales employ the bubble-net feeding strategy to capture their prey. A pod of whales encircles their target and creates a bubble barrier around them. The author in [26] examined the hunting behavior of baleen whale species by utilizing multi-sensor tags. Two separate movements of whales, that is, upward spirals and double loops, were recognized from his investigation. In upward spirals, the whale glides $12m$ below and generates a bubble in a spiral style around the prey and swims up toward the surface; however, in a double loop, the whale goes around the prey in three separate patterns: coral loop, lobtail, and capture loop. However, the original WOA has the following drawbacks:

1) Randomization has a key function in the initialization, exploration, and exploitation stages of WOA. So employing the existing randomization strategy in WOA would increase computational time, especially for the very complicated problem [28].
2) A single control parameter, a, affects both convergence and speed. This setting has a substantial effect on WOA performance. As a result, WOA has a modest convergence rate during both the exploration and exploitation phases. As a result, a proper balance between exploration and exploitation is essential [29].
3) In the search space, WOA uses the encircling mechanism, which has limited ability to jump out of local optima. As a result, it produces poor performance [30].

To address the shortcomings mentioned above, researchers have proposed several variants of it. WOA variants are mainly developed using two main approaches, improvement and hybridization [30]. In the improvement approach, standard techniques such as opposition based learning [31], Lamarckian learning [32], quadratic interpolation [33] chaotic map [1], single dimensional swimming [34], Laplace crossover operator [35] are used by the researchers to improve the WOA search strategies. In the hybridization approach, techniques such as differential evolution, evolutionary, genetic algorithm, sine-cosine algorithm [36], Grey wolf optimizer [37], and particle swarm optimization (PSO) [19] are used by the researchers to improve WOA performance. Table 1 shows a few recent variants of WOA. In general, the choice of a metaheuristic optimization algorithm variant is determined by several criteria, including the nature of the problem, performance goals, algorithm characteristics, implementation limitations, the possibility for hybrid or adaptive methods, and the search space [38] In [39], the authors have proposed a number of variants of WOA incorporating chaotic maps, such as: logistic, cubic, sine, tent etc.

In this research, it was discovered that the hybridization of logistic, sine, and tent maps with exponential maps helped further to improve the performance of the whale optimization algorithm.

2.1 Operation Of WOA

The algorithm operates through a sequence of three distinct stages: i) prey detection, ii) prey encirclement, and iii) prey capture. Humpback whales exhibit two distinct swimming patterns while pursuing prey: a linear route that gradually narrows or a spiral movement. The choice between these two motions is determined by a probability factor, p, which can flip between them. A balance between exploration and exploitation is maintained by $|\vec{A}|$ vector, which declines from 2 to 0 during iterations. In the early phase, when $|A| \geq 1$, the whales explore about the random prey, whereas as $|A| < 1$, the whales utilize the search space and swim around the best prey. Figure 1 shows the principle of WOA.

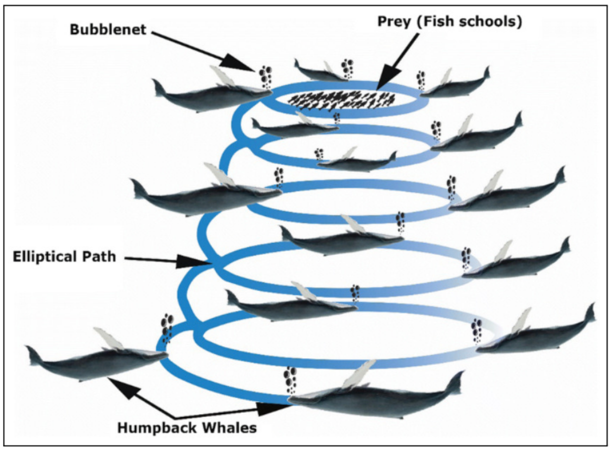

Fig.1: *Principle of Whale Optimization Algorithm [50].*

The mathematical model inspired by the spiral bubble-net feeding movement of whales around prey is outlined as follows:

Step 1: Initiate the population of whales (search agents) randomly within defined space:

$$X_i = (x_i \ldots\ldots\ldots x_n) \tag{1}$$

Where n signifies the space dimension.

Step 2: Evaluate the cost of each whale, and depending on the problem (minimization or maximization), find the position of the best whale $(X*)$

Table 1: *Recent variants of WOA.*

Ref.	*Method*	*Problem Addressed*	*Modifications*
[40]	MultiModal WOA (MMWOA)	Difficulty in finding multiple remedies	Clustering using k-means and fixed size using random number
[39]	Chaotic WOA	unbalance between exploitation and exploration phase	The initial population is generated using chaotic motion, non-linear control parameter $\overrightarrow{a}$ is used as a control parameter
[41]	Random hopping update strategy (PWOA) and random control parameter strategy (AWOA)	Poor convergence rate due to problems in finding global optima	Due to the position update strategy, new individuals are randomly generated near the optimal individuals, and due to random control parameters the convergence factor can non-linearly adjust the ability of global search and local search
[42], [43]	Opposition-Based WOA (OWOA)	Due to the random numbers used for initial population generation, they can be either near-to-optimum value or can be very far away from the optimal value. Hence the conversion time increases.	Random numbers and their inverses are employed concurrently to determine the optimal position of initial search agents. (concept of opposition-based learning (OBL))
[42], [44]	Opposition and Exponential WOA (OEWOA)	Unbalance between the exploitation and exploration phase	This version of WOA incorporates the concept of Origin-Based Linguistics (OBL) to enhance the diversity among search agents and thereby boost exploration. The exponentially decreasing function has been employed to enable the agents to thoroughly explore in the early iterations and utilize the search regions towards the conclusion.
[42]	Re-initialization Based WOA (RIWOA)	Unbalance between the exploitation and exploration phase	Worst particles are removed from the population and then re-initialized in the search space, and an exponentially decaying function is used for parameter $\overrightarrow{a}$. The particles whose fitness value is greater than the median of population are categorized as the worst particles.
[45]	WOA with joint search mechanism (JSWOA)	impoverished global search capability in the early stage, sluggish convergence rate at the later stage	Chaos theory was applied to improve the quality of the initial population position, adaptive inertia weight was used to enhance the convergence accuracy and speed, and the opposition-based learning (OBL) strategy was used to poise the exploration and exploitation of WOA.
[46]	self-adaptive whale optimization algorithm (SAWOA)	low precision, slow convergence, and trapped into local optimum due to the shortage of population diversity	The random number used to select the number of agents was multiplied by a function τ, which was found using fitness obtained in the current and previous iteration
[47], [48]	WOA based on Lamarckian learning (WOALam)	Problems in finding the global optimal solution of high-dimensional complex problems, problems of slow convergence	Based on Lamarck's evolutionary theory, individuals with more development potential are selected to perform a locally enhanced search.
[33]	whale optimization algorithm based on quadratic interpolation (QIWOA)	slow convergence rate, low solution accuracy, lack of diversity, and falling into local optimal for high dimension optimization problems	Quadratic interpolation (QI) technique is used. QI is a local search operator which uses a parabola to fit the shape of a quadratic function to obtain the intense point of the curve. This local search method has considerable reference value for WOA. The interpolation helps QIWOA search more efficiently around the optimal solution.
[39]	Chaotic WOA	unbalance between exploitation and exploration phase	The initial population is generated using chaotic motion, non-linear control parameter $\overrightarrow{a}$ is used as a control parameter
[49]	whale optimization algorithm based on Levy flight	Inconsistent exploration problem, that causes a problem of trapping of local optima	Levy Flight was utilized in the exploration phase of WOA to optimize the search agent's diversification

Step 3: Update the constant parameters A and C using the following equations:

$$A = 2\bar{a} \cdot \bar{r} - \bar{a} \tag{2}$$

$$C = 2 \cdot \bar{r} \tag{3}$$

where $\bar{r}$ is the random number in the range $[0, 1]$ and $\bar{a}$ is iteratively decreased from 2 to 0

Step 4: If $p < 0.5$ and $|A| > 1$, then select the random position of the whale (X_{rand}) in search space and update the position of the whale around it using the following equations:

$$D = |C * X_{rand} - X| \tag{4}$$

$$X(t+1) = X_{rand} - A * D \tag{5}$$

Else if, $p < 0.5$ and $|A| < 1$, then update the position of the whale around the best search agent (X^*) using the following equation:

$$D = |C * X^* - X| \tag{6}$$
$$X(t+1) = X^* - A * D \tag{7}$$

Else, if $p > 0.5$, then update the position of the whale using the following equation:

$$X(t+1) = D'.e^{bl}.\cos(2\pi l) + X^*(t) \tag{8}$$

where $D' = |X^*t - Xt|$ is the distance between the whale and best-searched prey ($X'(t)$), b is the constant that maintains the shape of the logarithmic spiral and l is the random number defined in the range $[-1, 1]$, is element-by-element multiplication [1].

Step 5: Re-initialize the position of whales that goes beyond the search space.

Step 6: The algorithm terminates when it achieves either the smallest error or the maximum number of iterations specified. Otherwise, repeat steps (2)–(6).

Step 7: The position of X^* represents the global optimal solution.

Algorithm 1 shows the pseudo code of WOA proposed by Mirjalili *et al.* in [26]. As per this algorithm "The WOA algorithm starts with a set of random solutions [26]. At each iteration, search agents update their positions concerning either a randomly chosen search agent or the best solution discovered so far. The a parameter is dropped from 2 to 0 to provide exploration and exploitation, respectively [26]. A random search agent is chosen when $|\vec{A}| > 1$, whereas the optimal solution is selected when $|\vec{A}| < 1$ for updating the position of the search agents. Depending on the value of p, WOA is able to flip between either a spiral or circular movement. Finally, the WOA algorithm is ended by satisfying a termination criterion" [26].

Algorithm 1: *Pseudo-code of the WOA [26].*

Initialize the whales' population X_i ($i = 1, 2, \ldots, n$)
Calculate the fitness of each search agent
X* = the best search agent
while $t <$ maximum number of iterations **do**
  for each search agent **do**
    Update a, A, C, l, and p
    if $p < 0.5$ **then**
      if ($|A| < 1$) **then**
        Update the position of the current search agent by Eq. 7
      else if ($|A \geq 1$) **then**
        Select a random search agent (X_{rand})
        Update the position of the current search agent by Eq. 4
      end if
    else if ($p \geq 0.5$) **then**
      Update the position of the current search agent by Eq. 8
    end if
  end for
  Check if any search agent goes beyond the search space and amend it
  Calculate the fitness of each search agent.
  Update X^* if there is a better solution.
  ($t = t + 1$)
end while
Return X^*

research trend suggests that substituting traditional pseudo-random number generators with chaotic maps might create more successful solutions to solve complicated optimization challenges [30]. Thus, to improve the WOA approach, researchers apply different chaotic maps such as logistic, sine, Chebyshev, circle, iterative, skew tent map, gauss/mouse, sinusoidal, and singer [30]. These chaotic maps are applied to create the initial population, producing values in control parameters and updating the position of search agents [53],[54]. A few chaotic maps and associated mathematical equations are shown in Table 2.

Table 2: *Chaotic maps and their equations.*

Sr.	*Chaotic Map Name*	*Chaotic Map Formula*
1	Logistic	$x_{i+1} = ax_i(1 - x_i)$,$a = 4$
2	Sine	$x_{i+1} = \frac{a}{4}\sin(\pi x_i)$
3	Circle	$x_{i+1} = \mod(x_i + b - \frac{a}{2\pi}\sin(2\pi x_k), 1)$
3	Iterative	$x_{i+1} = \sin(\frac{a\pi}{x_i}), a = 0.7$
4	Gauss/mouse	$x_{i+1} = \begin{cases} 1 & \text{if } x_i = 0, \\ \frac{1}{\mod(x_i, 1)}, & \text{otherwise.} \end{cases}$
5	Singer	$x_{i+1} = \mu(7.86x_i - 23.31x_i^2 + 28.75x_i^3 - 13.302875x_i^4), \mu = 1.07$
6	Cubic	$x_{i+1} = ax_i(1 - x_i^2)$

3. EXPONENTIAL CHAOTIC MAP

In this section we offer an overview of exponential chaotic maps. We have combined some of these exponential chaotic maps to introduce Exp-WOA method in later section.

3.1 Chaotic Maps

Mathematically, a chaotic map is a complicated, random phenomenon having a specific interior structure [51]. In chaotic systems, even minor changes in starting conditions can lead to significant future changes. This phenomenon is referred to as the butterfly effect in chaos theory [52]. Metaheuristic algorithms use randomness to diversify search processes and avoid local optimization. A new metaheuristics

3.2 Exponential Map

The exponential map was proposed by Zhongyun Hua *et al.* in [55].

As shown in Figure 2 the exponential map was constructed using base and exponent maps. The base map was represented using $f(a, x_i)$ and exponent map was represented using $g(b, x_i)$, a and b are their con-

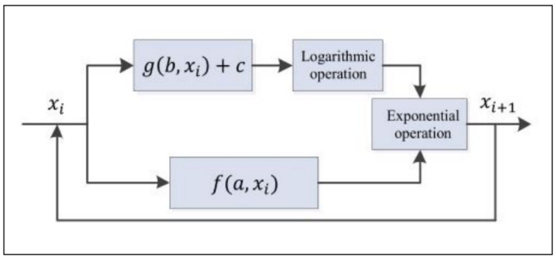

Fig.2: *Structure of Exponential Map Proposed in [55].*

trol parameters and c is a small bias to balance the output of $g(b, x_i)$ [55].

The mathematical expression for the exponential map is denoted as:

$$x_{i+1} = \mathcal{E}(§_\rangle) = \{(\dashv, §_\rangle)^{\ln(\}(\lfloor, §_\rangle)+\rfloor)} \quad (9)$$

In [55], the output of the exponential map is generated in iteration form. The exponential map has the following characteristics:

- Users have the ability to create a huge number of chaotic maps by combining the base map $f(a, x_i)$ with the exponent map $g(b, x_i)$ [55].
- The base map $f(a, x_i)$ and exponent map $g(b, x_i)$ might be equal or distinct one-dimensional chaotic maps [55].
- By simply altering the values of $f(a, x_i)$ and $g(b, x_i)$, completely alternative chaotic maps employing logistic, sine, or tent maps may be generated.

The logistic-exponent-logistic (LEL), logistic-exponent-sine (LES), logistic-exponent-tent (LET), sine-exponent-sine (SES), sine-exponent-logistic (SEL), sine-exponent-tent (SET), tent-exponent-sine (TES), tent-exponent-logistic (TEL), and tent-exponent-tent (TET) maps are proposed in [55]. We have employed logistic-exponential-logistic, logistic-exponential-sine, and logistic-exponential-tent map to propose Exp-WOA in this work, as presented in Table 3.

The bifurcation diagrams of these three exponential maps over the parameter (b, c) space are shown in Figure 3., where $b \in [0, 1]$ and $c \in [2, 2.8]$. These three chaotic maps exhibit complicated, chaotic behaviors in all parameter values, as can be seen. Their output values are uniformly dispersed across the entire range. Base and exponent maps display chaotic behavior in small parameter ranges and do not have equally distributed outputs [55]. These novel chaotic maps, with uniform-distribution outputs, are ideal for various applications, including pseudo-random number generators.

Table 3: *Chaotic maps produced using logistic and sine maps [55].*

Sr.	*Exponential chaotic map type*	*Mathematical formula*
1	logistic-exponent-logistic	$x_{i+1} = (4ax_i(1-x_i))^{\ln(4bx_i(1-x_i)+c)}$
2	logistic-exponent-sine	$x_{i+1} = (4ax_i(1-x_i))^{\ln(b\sin(\pi x_i)+c)}$
3	logistic-exponent-tent	$x_{i+1} = (4ax_i(1-x_i))^{\ln(2b\min\{x_i,1-x_i\}+c)}$

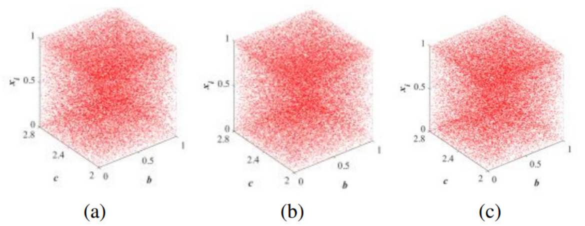

Fig.3: *Bifurcation diagrams of (a): LEL map (b): LES map (c): LET map. [55].*

4. EXPONENTIAL WOA (EXP-WOA)

In this section, we propose three ways to avoid sluggish convergence and the tendency to slip into local optimum solutions by combining LEL, LES, and LET exponential maps with WOA. These three approaches are: i) LEL-WOA, ii) LES-WOA, and iii) LET-WOA. Despite having a high convergence rate, WOA cannot outperform other algorithms in discovering global optima, which affects the algorithm's convergence rate. To mitigate this effect and increase its efficiency, the Exp-WOA algorithm is developed by including exponential chaotic maps within the WOA algorithm itself. In general, chaotic is derived from the word 'chaos', which refers to the property of a complex system whose behavior is unpredictable, and map refers to the process of mapping or correlating chaos behavior in an algorithm with some parameter via a function. Because of the ergodicity and non-repetition features of chaos, it can execute overall searches faster than stochastic searches, which rely on probabilities [56].

fig:4 shows the flow-chart of Exp-WOA. The initial section of the flow chart involves the initialization of different parameters of the WOA. Subsequently, the whale population has been initiated using the LEL, LES, or LET chaotic map. The number of chaotic maps is starting to vary the $'p'$ parameter of the method, leading to chaos [1]. During the following stage, the total fitness of every whale inside the search region is examined using multiple defined benchmark functions. The whale displaying the highest degree of physical condition is regarded as the most efficient search agent presently. The optimum search agent will consistently update its position using Equation 4, as long as the control parameter A is less than 1. Moreover, when the parameter A exceeds 1, a random whale is selected [1], and the position of the current top-performing search agent is adjusted using LEL,

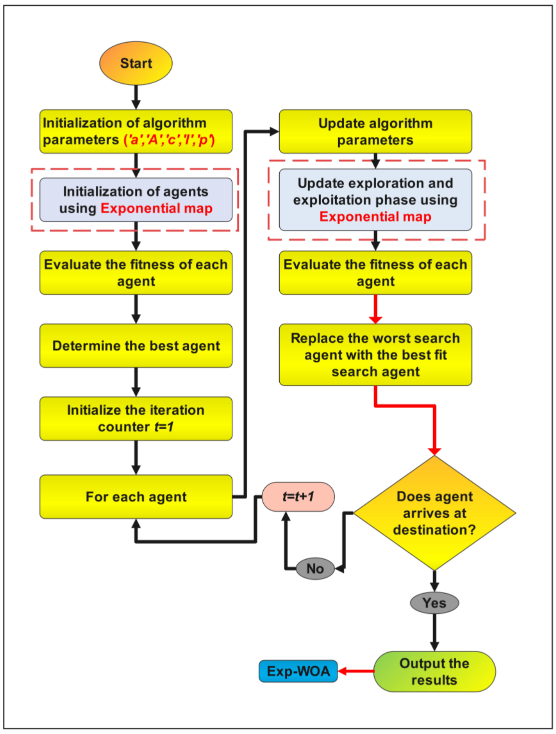

Fig.4: *Flowchart of Optimization Procedure of Exp-WOA.*

Algorithm 2: *Pseudo-code of Exp-WOA algorithm.*

```
1: Initialize the whales population X_i using LEL OR LES OR LET exponential chaotic maps
2: Calculate the fitness of each search agent
3: X* = the best search agent
4: while t < maximum number of iterations do
5:    for each search agent do
6:       Update a, A, C, l, and p
7:       if p < 0.5 then
8:          if (|A|< 1) then
9:             Update the position of the current search agent by the Eq. 7
10:         else if (|A ≥ 1) then
11:            Select a random search agent (X_rand)
12:            Update the position of the current search agent using LEL OR LES OR LET chaotic maps by Eq. 4
13:         end if
14:      else if (p ≥ 0.5) then
15:         Update the position of the current search agent by the Eq. 8
16:      end if
17:   end for
18:   Check if any search agent goes beyond the search space and amend it
19:   Calculate the fitness of each search agent.
20:   Update X* if there is a better solution.
21:   (t = t + 1)
22: end while
23: Return X*
```

LES, or LET chaotic maps created in step 2, but only if a new search agent surpasses the previous one. The very versatile whale will sequentially revise its location and may finally select the initial position as the most advantageous alternative. The parameter $'p'$ is modified by the number of iterations, utilizing equations 2 and 3. The Exp-WOA algorithm will consider the best search agent as the most optimum solution at the conclusion of the last iteration. The pseudo-code of the proposed Exp-WOA algorithm is shown in Algorithm 2. The mathematical analysis of exponential maps with WOA has been presented in the next sub-section.

4.1 Mathematical Analysis Of Exponential Maps

Instead of initializing the population randomly, the proposed maps are used to generate the initial population as shown in the following Equation:

$$x_i(0) = LB + (UB - LB) \cdot ChaoticSequence(i) \tag{10}$$

where,

- LB and UB are the lower and upper bounds of the search space
- ChaoticSequence(i) is generated using the proposed exponential chaotic maps

Then, in the position update steps of WOA, the chaotic sequence is incorporated to enhance the randomness and diversity:

$$\begin{aligned} x_i(t+1) =& x_{best}(t) + ChaoticSequence(t) \\ & \cdot (x_i(t) - x_{best}(t)) \end{aligned} \tag{11}$$

where,

- $x_i(t)$ is the position of i^{th} whale at iteration t
- $x_{best}(t)$ is the position found so far
- $ChaoticSequence(t)$ is the sequence generated using the proposed exponential chaotic map.

The logistic component in the proposed maps ensures that small changes in initial conditions lead to significantly different trajectories, promoting exploration. The sinusoidal component introduces periodic oscillations, helping the algorithm explore the search space in diverse directions. The Tent component adds piecewise linearity, promoting better space-filling properties. Exponential chaotic maps provide more varied and broader initial positions due to the non-linear exponential term, allowing the algorithm to more effectively traverse the search space. Furthermore, exponential chaotic maps have vital space filling properties due to the exponential term, guaranteeing that the search space is covered without noticeable gaps. During the update phase, these sequences assist in fine-tuning positions by limiting premature convergence to local optima [57]. Furthermore, due to above-mentioned properties, these sequences pro-

Table 4: *Benchmark functions.*

No.	*Function name*	*Mathematical equation*	Dim	Range	Extreme value
F1	Sphere	$f_x = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
F2	Beale	$f_x = (1.5 - x_i + x_i x_{i+1})^2 + (2.25 - x_i + x_i x_{i+1}^2)^2 + (2.625 - x_i + x_i x_{i+1}^3)^2$	2	$[-4.5, 4.5]$	0
F3	Cigar	$f_x = x_1^2 + \sum_{i=2}^{n} x_i^2$	30	$[-100, 100]$	0
F4	Step	$f_x = \sum_{i=0}^{n-1} (\lvert x\rvert + 0.5)^2$	30	$[-100, 100]$	0
F5	Quartic Noice	$f_x = \sum_{i=0}^{n-1} x_i^4 + N(0.1)$	30	$[-1.28, 1.28]$	0
F6	Bohachevsky	$f_x = x_i^2 + 2.0x_{i+1}^2 - 0.3\cos(3\pi x_i)\cos(4\pi x_{i+1}) + 0.7$	2	$[-100, 100]$	0
F7	Ackley	$f_x = -20exp(0.02)\sqrt{1}/D \sum_{i=1}^{d} \cos(2\pi x_i) + 20 + exp$	30	$[1.28, 1.28]$	0
F8	Griewank	$f_x = \frac{1}{4000} \sum_{i=0}^{n-1} (x_i - 100)^2 - \prod_{i=1}^{n-1} \cos(\frac{x_i - 100}{\sqrt{r-1}}) + 1$	30	[-500,+500]	0
F9	Levy	$f_x = \sin 2(\pi\omega 1) + \sum_{i-1}^{d-1} (\omega i - 1)2[1 + 10\sin 2(\pi\omega i + 1)] + (\omega d - 1)2[1 + \sin 2(2\pi\omega d)]$	30	$[-5.12, 5.12]$	0
F10	Michalewitz	$f_x = -\sum_{i=0}^{n} (\sin(x_i)) \sin^{20}(\frac{i_x i^2}{\pi})^{2m}$	30	$[-32, 32]$	0.96
F11	Rastrigin	$f_x = (x_i^2 - 10\cos 2\pi x_i + 10)$	30	$[-600, 600]$	0
F12	Alpine	$f_x = \sum_{i=0}^{n} \lvert x_i \sin x_i + 0.2x_i \rvert$	30	$[-50, 50]$	0
F13	Schaffer	$f_x = (x_i^2 + x_{i+1}^2)^{\frac{1}{4}} (50(x_i^2 + x_{i+1}^2)^{0.1} + 1)$	30	$[-50, +50]$	0
F14	Rosenbrock	$f_x = 100(x_{i+1} - x_i^2)^2 + (1.0 - x_i)^2$	30	$[-65.536, +65.536]$	0
F15	Easom	$f_x = \cos(x_i)\cos(x_{i+1})\exp(-(x_i - \Omega)^2 - (x_{i+1} - \Omega)^2)$	4	$[-5, +5]$	0
F16	Shubert	$f_x = (\sum_{i=1}^{5} i\cos(i+1)x_{i+1} + i)$	2	$[-5, 5]$	0
F17	Schwefel 1.2	$f_x = \sum_{i=0}^{n-1} \{\sum_{j=0}^{j<1} x_i\}2$	2	$[-10, 10]$	0
F18	Schwefel 2.21	$f_x = \max(\lvert x_i \rvert)$	30	$[-10, 10]$	0
F19	Schwefel 2.22	$f_x = \sum_{i=0}^{n-1} \lvert x_i \rvert + \prod_{i=1}^{n-1} \lvert x_i \rvert$	30	$[-10, 10]$	0
F20	Schwefel 2.26	$f_x = -\sum_{i=0}^{n-1} x_i \sin\sqrt{x_i}$	30	$[-10, 10]$	0
F21	-	$f_x = -\sum_{i=1}^{5} [(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.15
F22	-	$f_x = -\sum_{i=1}^{7} [(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.40
F23	-	$f_x = -\sum_{i=1}^{10} [(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.53

Table 5: *Comparison of optimization results obtained for the 23 Benchmark functions.*

F	PSO		WOA		L-WOA		S-WOA		T-WOA		LEL-WOA		LES-WOA		LET-WOA	
	avg	std	avg	std	avg	std	avg	std	avg	std	avg	std	avg	std	avg	std
F1	7.67E-63	3.54E-62	4.33E-69	2.33E-68	2.453E-69	3.545E-62	3.574E-67	2.542E-65	2.386E-67	3.331E-56	1.4E-70	4.40E-69	1.763E-66	3.509E-62	1.655E-61	3.897E-60
F2	5.76E-48	2.16E-43	6.2E-51	2.49E-50	5.89E-49	2.11E-50	5.564E-45	1.95E-52	5.125E-41	2.76E-50	5.08E-36	9.18E-60	4.342E-55	8.66E-38	4.95E-45	8.59E-48
F3	6.5E+5	2.29E+5	4.36E+4	1.30E+4	4.54E+3	1.14E+4	4.77E+3	1.85E+4	4.55E+5	1.97E+4	4.10E+3	0.6E+4	4.56E+5	2.76E+4	4.73E+5	2.82E+4
F4	4.877E+1	3.769E+2	4.526E+1	2.743E+1	4.376E+1	2.287E+1	7.65E+1	5.062E+1	4.454E+1	2.359E+1	3.983E+1	2.259E+1	3.765E+1	2.189E+1	3.553E+1	2.008E+1
F5	4.732E+2	5.875E+2	2.811E+1	5.02E-1	2.543E+1	3.26E-1	2.767E+1	5.9E-1	2.765E+11	4.36E-1	2.356E+1	3.76E-1	5.554E+1	9.94E-1	1.976E+1	3.38E-1
F6	4.769E+1	2.953E+0	4.22E-1	2.22E-1	5E-3	1.36E-1	1.28E-1	5E-3	4.74E-1	3.64E-1	2.81E-1	2.11E-1	6E-4	1.55E-1	4.65E-1	3.95E-1
F7	4.076E-2	6.54E-2	3.9E-3	5.2E-3	1.5E-3	4.8E-3	6.72E-1	4.37E-1	4E-5	3E-4	2.1E-3	1.5E-3	4.7E-3	8.9E-3	2.7E-3	3E-4
F8	-7.76E+4	19.54E+4	-10.582E+3	16.49E+2	-11.234E+2	12.17E+2	-25.57E+5	19.76E+2	-9.76E+4	18.55E+2	-12.39E+4	6.13E+2	-10.665E+2	12.673E+3	-5.675E+3	12.669E+2
F9	0.001+E0	0.055E+1	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0	0.00E+0	5.68E-15	2.24E-14	3.65E-12	1.5E-14	0.00E+0	0.00E+0
F10	2.347E-13	2.89E-12	3.40E-15	2.44E-14	2.85E-13	2.71E-13	2.96E-14	2.37E-14	5.1E-16	2.24E-15	5.77E-15	3.00E-15	4.89E-16	2.76E-18	4.38E-17	2.97E-18
F11	0.003E-1	0.005E+1	0.00E+0	0.00E+0	1.5E+0	21.55E+2	0.78E+2	17.65E+2	0.896E+2	54.76E+2	3.70E-18	1.990E-17	26.67E+2	56.47E+2	45.84E+2	56.32E+2
F12	4.65E-1	3.617E-1	2.0E-2	1.6E-2	5.6E-2	2.6E-2	7.3E-2	6.5E-2	5.7E-2	7.1E-2	6.3E-2	3.4E-3	4.5E-3	9.2E-2	9.8E-2	2.30E-1
F13	6.76E+0	4.62E+0	5.09E-1	2.089E-1	6.86E+0	1.87E+0	5.56E+0	7.3E-1	6.94E+0	3.9E-1	1.24E-1	6.0E-3	4.06-1	1.99E-1	4.53E-1	1.73E-1
F14	2.134E+1	2.26E+1	2.277E+0	2.422E+0	5.127E+0	4.67E+0	3.19E+0	2.78E+0	3.13E+0	5.69E+0	1.165E+0	4.49E-1	2.07E+0	1.87E+0	2.96E+0	1.93E+0
F15	1.56E-2	7.87E-2	1.03E-3	6.0E-4	1.01E-3	7.87E-4	1.25E-4	5.77E-4	1.996E-3	4.34E-4	1.01E-3	8.25E-5	1.25E-3	5.88E-4	1.01E-4	5.77E-4
F16	7.5E+0	1.34E+0	5.09E-1	2.089E-1	6.76E-1	1.67E+0	4.96E-1	2.3E-1	1.54E-1	2.39E-1	5.4E-2	1.53E-1	8.7E-2	1.64E-1	4.83E-1	2.72E-1
F17	5.34E+0	2.96E-3	3.978E-1	2.7E-5	5.54E-1	5.76E-5	13.78E+0	6.43E-4	8.77E-1	2.1E-5	3.979E-1	2.0E-4	2.73E-1	2.18E-6	2.54E-1	3.67E-6
F18	1.129E+1	0.565E-3	3.00009E+0	1E-4	7.76E+0	4E-4	2.81E+0	0.75E-4	2.21E+0	0.97E-4	3.00003E+0	7.888E-05	2.419E+0	5.73E-5	3.55E+0	2.19E-4
F19	-2.963E+0	9.76E-3	-3.856E+0	6.8E-3	6.43E+0	1.55E+0	-3.65E+0	4.63E-2	-2.93E+0	4.34E-4	-3.240E+0	4.13E-2	-8.68E+0	17.85E-5	-5.56E+1	5.82E-4
F20	-1.495E+0	2.92E+0	-3.206E+0	2.295E-1	-2.67E+0	9.3E-1	-5.74E+0	2.5E-1	-11.74E-0	2.25E-1	-3.1911E+0	1.007E-1	-17.62E+0	1.39E-1	-7.26E+0	1.93E-1
F21	-6.564E+0	3.378E+1	-8.080E+0	2.4717E+0	-6.63E+0	3.59E+0	-15.87E+0	1.95E+0	-10.84E+0	2.21E+0	-25.003E+0	1.415E-1	-15.76E+0	2.14	-9.65E+0	1.79E+0
F22	-5.804E+1	4.429E+1	-7.224E+0	3.2611E+0	-16.76E+0	2.55E+0	-14.76E+0	1.76E+0	-10.69E+0	3.17E+0	-29.2360E+0	2.167E+0	-15.84E+0	1.39E+0	-19.32E+0	1.13E+0
F23	-4.843E+0	4.67E+1	-6.833E+0	3.4096E+0	-28.64E+0	5.0E-2	-19.53E+0	1.42E+0	-10.39E+0	1.74E+0	-10.409E+0	1.23E-0	-21.64E+0	3.156E+0	-25.74E+0	2.61E+0

vide a balance between randomness and deterministic patterns, enhancing the algorithm's capacity to avoid local minima and converge to global optima.

5. RESULTS AND DISCUSSION

Every novel optimization approach must deal with several mathematical test functions to be examined and tested. Several experiments on optimization benchmark issues are carried out in this portion to validate the performance of the proposed Exp-WOA algorithms. A population size of 30 and a maximum iteration equal to 500 have been utilized. Twenty-three well-known benchmark functions were used to test Exp-WOA's performance. These functions are characterized as either unimodal or multimodal [26]. Unimodal benchmark functions contain a single optima and are particularly suited for benchmarking [1]. Multimodal benchmark functions, on the other hand, have more than one optima, making them more complex than unimodal functions [1]. One optima is known as the global optima, while the others are known as local optima [26]. The main qualities of any robust meta-heuristic algorithm should be avoiding local optima and determining the global optimum [1]. As a result, multimodal benchmark functions are in charge of testing exploration and preventing entrapment in local optima [1]. Table 4 defines the test functions that reflect the *cost* function, *Dim* specifies the dimension of the function, the *range* represents the range of variation of optimization variables, and the *extreme value* represents the final optimal value. In this section, Exp-WOA was also compared to various non-exponential map based approaches.

5.1 Performance Comparison Of Exp-WOA

The experiments conducted with Exp-WOA involve a population size of 30 whales and 500 iterations. The outcomes are subsequently computed by taking the average of thirty separate rounds. Logistic-WOA (L-WOA), sinusoidal-WOA (S-WOA), tent-WOA (T-WOA), LEL-WOA, LES-WOA, LET-WOA utilize to logistic, sine, tent, logistic-exponential-logistic, logistic-exponential-sine and logistic-exponential-tent maps, respectively, as shown in Table 2 and Table 3. It can be seen from Table 5 that LEL-WOA performs better for 10 benchmark functions (F1, F3, F7-F8, F13-F16, F21-F22), LES-WOA performs better for 6 benchmark functions (F2, F6, F10, F12, F18, F20) and LET-WOA performs better for 5 benchmark functions (F4, F5, F17, F19, F23) as compared to WOA. In other words, exponential maps such as logistic-exponential-logistic, logistic-exponential-sine, and logistic-exponential-tent maps can enhance the effectiveness of the WOA algorithm. From the data in Table 5, it is proven that the exponential-based WOA algorithm delivers the best results on all test functions except for F9 and F11. It can also been seen from Table 5 that, the proposed approaches outperforms the state of the art algorithm, such as particle swarm optimization (pso) algorithm [58] for almost all benchmark functions.

5.2 Qualitative Analysis

A qualitative study has been conducted on several benchmark functions to evaluate the performance of Exp-WOAs further. Figs. 5-10 indicate the co-relation between convergence speed of proposed methods (LEL-WOA, LES-WOA, and LET-WOA) with the existing methods (WOA, L-WOA, S-WOA, and T-WOA). The convergence curves of Exp-WOAs have been displayed across 500 iterations to facilitate straightforward observation and analysis. Table 5 indicates that LEL-WOA, LES-WOA, and LET-WOA outperform other approaches on twenty-one benchmarks for function minimum search. L-WOA, S-WOA, and T-WOA are the second best algorithms, performing best on fifteen out of twenty-three benchmark functions. These data represent average function optimum values obtained from thirty runs. Here, all the values represent accurate function values.

Figs. 5-10 exhibits the data collected by exponential maps, and other maps such as logistic, sinusoidal and tent on various mathematical functions. According to Fig. 6, LEL-WOA suppresses all other approaches for the F1 function, and according to Fig.

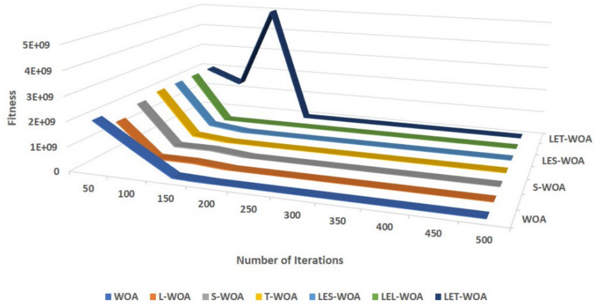

Fig.5: *Performance comparison on the F1 Sphere function.*

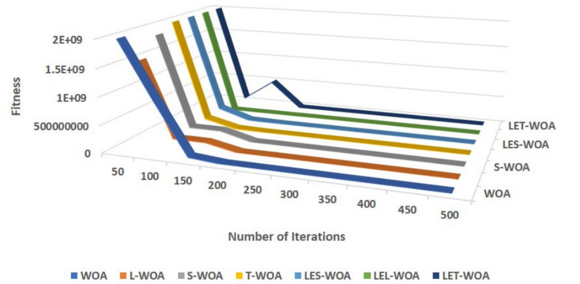

Fig.6: *Performance comparison on the F4 Step function.*

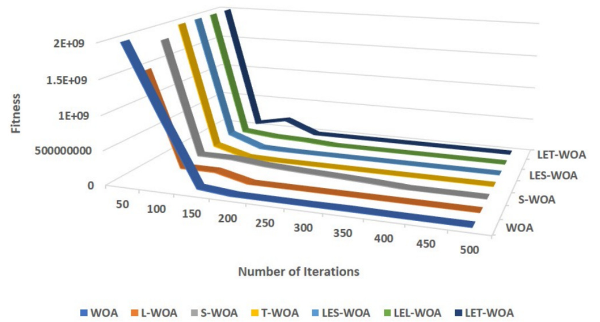

Fig.7: *Performance comparison on the F6 Bohachevsky function.*

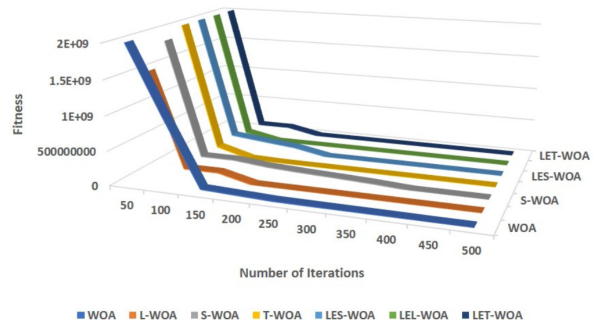

Fig.8: *Performance comparison on the F8 Griewank function.*

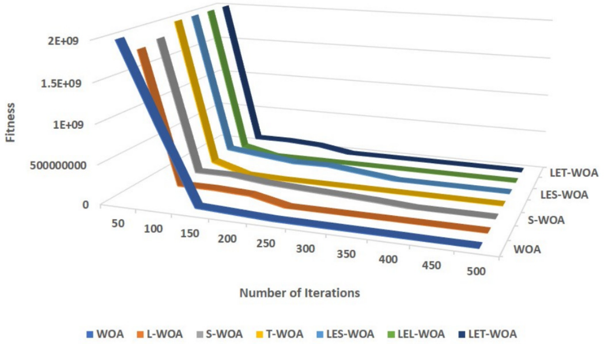

Fig.9: *Performance comparison on the F12 Alpine function.*

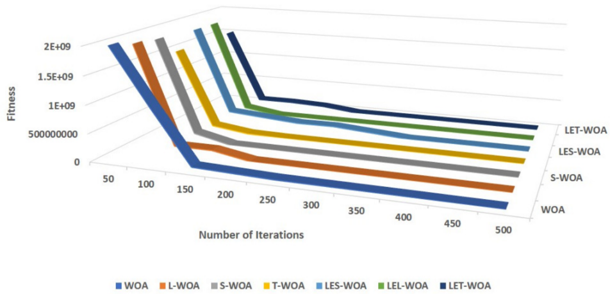

Fig.10: *Performance comparison on the F20 Schwefel 2.26 function.*

7, LEL-WOA has the quickest convergence rate for the F4 function. According to Fig. 8, LES-WOA suppresses all other approaches for the F6 function, and according to Fig. 8, LES-WOA, and LET-WOA show the fastest convergence rate and overtake all other methods for the function F8. According to Fig. 9, LES-WOA converges faster for the function F12, and according to Fig. 10, LET-WOA shows a faster convergence rate and overtakes all other methods for the function F20.

Testing a metaheuristic optimization method with unimodal, multimodal, and continuous mathematical functions is one of the criteria for determining its validity and correctness [59]. From Table 5 and Figs. 5-10 it can be concluded that LEL-WOA and LES-WOA perform better in terms of their convergence speed as compared with WOA, L-WOA, S-WOA, and T-WOA on twenty-one benchmark functions out of twenty-three.

5.3 Time Complexity Of Proposed Methodologies

The time complexity of the algorithms under study for F1-F23 benchmark functions has been presented in Table 6. The time complexity of all the proposed algorithms using exponential maps is higher than that of the original WOA for nearly all mathematical benchmark functions, as shown in Table 6. The increased time complexity of the whale optimization algorithm integrated with chaotic maps compared to its basic form arises from additional computational steps, the complexity of chaotic systems, enhanced search mechanisms, the generation of chaotic sequences, and the need for parameter tuning. While these additional computations can improve the algorithm's performance in discovering global optima, they also lead to a more considerable computational cost.

Table 6: *Time complexity (in seconds) of the algorithms under study.*

Function	WOA	L-WOA	S-WOA	T-WOA	LEL-WOA	LES-WOA	LET-WOA
F1	1.62E-1	1.43E-1	1.25E-1	1.15E-1	3.52E-1	5.43E-1	6.75E-1
F2	1.35E-1	1.74E-1	1.45E-1	1.95E-1	5.74E-1	6.61E-1	7.94E-1
F3	5.77E-1	4.32E-1	6.15E-1	7.28E-1	6.18E-1	6.74E-1	7.12E-1
F4	4.41E-1	5.17E-1	5.92E-1	6.39E-1	6.57E-1	5.12E-1	7.27E-1
F5	1.93E-1	2.85E-1	2.63E-1	2.91E-1	3.44E-1	2.17E-1	3.32E-1
F6	1.19E-1	1.71E-1	2.745E-1	2.59E-1	1.72E-1	2.24E-1	4.33E-1
F7	2.48E-1	2.37E-1	2.10E-1	1.17E-1	2.48E-1	1.82E-1	4.37E-1
F8	1.02E-1	1.18E-1	1.94E-1	1.015E-1	2.21E-1	2.29E-1	2.05E-1
F9	1.09E-1	1.28E-1	1.99E-1	1.26E-1	2.02E-1	3.14E-1	2.07E-1
F10	1.72E-1	1.84E-1	1.94E-1	2.71E-1	3.17E-1	4.17E-1	4.45E-1
F11	2.23E-1	2.48E-1	2.41E-1	2.98E-1	2.99E-1	3.07E-1	2.39E-1
F12	5.75E-1	5.93E-1	6.73E-1	7.64E-1	6.41E-1	6.10E-1	7.27E-1
F13	4.99E-1	5.33E-1	5.98E-1	5.65E-1	5.02E-1	6.21E-1	6.94E-1
F14	6.75E-1	6.92E-1	6.85E-1	7.05E-1	7.31E-1	7.27E-1	6.19E-1
F15	5.42E-1	5.12E-1	5.73E-1	4.15E-1	4.18E-1	4.13E-1	4.11E-1
F16	2.96E-1	3.07E-1	3.76E-1	3.52E-1	3.19E-1	4.15E-1	5.19E-1
F17	2.52E-1	2.04E-1	2.06E-1	2.31E-1	2.14E-1	2.38E-1	2.08E-1
F18	7.55E-1	7.40E-1	6.99E-1	6.925E-1	6.71E-1	8.49E-1	7.96E-1
F19	1.85E-1	1.97E-1	2.05E-1	2.32E-1	2.98E-1	3.01E-1	3.23E-1
F20	2.25E-1	2.653E-1	3.05E-1	3.10E-1	3.34E-1	3.19E-1	3.18E-1
F21	9.74E-1	10.63E-1	10.23E-1	10.16E-1	9.96E-1	10.12E-1	10.45E-1
F22	10.92E-1	10.05E-1	11.20E-1	11.57E-1	11.42E-1	11.85E-1	11.81E-1
F23	10.54E-1	11.15E-1	11.58E-1	12.06E-1	11.93E-1	12.50E-1	12.87E-1

6. CONCLUSION AND FUTURE SCOPE

This study presents three variations of WOA that use chaotic maps: LEL-WOA, LES-WOA, and LET-WOA. These variants use an exponential map, a sine map, and a tent map, all combined with WOA. The performance of these three variations was evaluated against twenty-three benchmark functions. The experimental results reveal that LEL-WOA and LES-WOA outperform the original WOA, L-WOA, S-WOA, T-WOA, and PSO algorithms regarding convergence time and fitness value. For a few benchmark functions, LET-WOA performs marginally worse than other methods. The experimental results reveal that the proposed approaches have a larger time complexity than the original WOA and the other chaotic map-based approaches presented in this research. The chaos generated by the chaotic maps during the initial phase is the primary rationale behind Exp-WOA's superior performance. The chaos helps to manage better the exponential and exploratory phases, which increases the algorithm's convergence rate. In the future, it would be exciting to apply the Exp-WOA algorithm to real-world engineering problems.

AUTHOR CONTRIBUTIONS

Conceptualization, Vaibhav Godbole and Dr. Shilpa Gaikwad; methodology, Vaibhav Godbole; coding in Matlab, Vaibhav Godbole; selection of mathematical functions for validation, Vaibhav Godbole and Dr. Shilpa Gaikwad; formal analysis, Vaib-

hav Godbole; investigation, Vaibhav Godbole; writing—original draft preparation, Vaibhav Godbole; writing—review and editing, Vaibhav Godbole and Dr. Shilpa Gaikwad; visualization, Vaibhav Godbole and Dr. Shilpa Gaikwad; supervision, Vaibhav Godbole. All authors have read and agreed to the published version of the manuscript.

References

[1] G. Kaur and S. Arora, "Chaotic whale optimization algorithm," *Journal of Computational Design and Engineering*, vol. 5, no. 3, pp. 275–284, 01 2018. [Online]. Available: `https://doi.org/10.1016/j.jcde.2017.12.006`

[2] M. O. Okwu and L. K. Tartibu, *Metaheuristic Optimization: Nature-Inspired Algorithms Swarm and Computational Intelligence*, Theory and Applications. Springer Nature, 2020, vol. 927.

[3] M. J. Heule and O. Kullmann, "The science of brute force," *Communications of the ACM*, vol. 60, no. 8, pp. 70–79, 2017.

[4] J. Edmonds, "Matroids and the greedy algorithm," *Mathematical programming*, vol. 1, pp. 127–136, 1971.

[5] S. Forrest, "Genetic algorithms," *ACM computing surveys (CSUR)*, vol. 28, no. 1, pp. 77–80, 1996.

[6] P. Sharma and S. Raju, "Metaheuristic optimization algorithms: A comprehensive overview and classification of benchmark test functions," *Soft Computing*, pp. 1–64, 2023.

[7] A. Ravindran, G. V. Reklaitis, and K. M. Ragsdell, *Engineering optimization: methods and applications*. John Wiley & Sons, 2006.

[8] X. Fan, W. Sayers, S. Zhang, Z. Han, L. Ren, and H. Chizari, "Review and classification of bio-inspired algorithms and their applications," *Journal of Bionic Engineering*, vol. 17, pp. 611–631, 2020.

[9] S. Y. Ko, I. Gupta, and Y. Jo, "Novel mathematics-inspired algorithms for self-adaptive peer-to-peer computing," in *First International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2007)*. IEEE, pp. 3–12, 2007.

[10] A. Biswas, K. Mishra, S. Tiwari, and A. Misra, "Physics-inspired optimization algorithms: a survey," *Journal of Optimization*, vol. 2013, 2013.

[11] O. D. Montoya, A. Molina-Cabrera, and W. Gil-González, "A possible classification for metaheuristic optimization algorithms in engineering and science," *Ingeniería*, vol. 27, no. 3, 2022.

[12] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "Gsa: a gravitational search algorithm," *Information sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[13] A. Hatamlou, "Black hole: A new heuristic optimization approach for data clustering," *Information sciences*, vol. 222, pp. 175–184, 2013.

[14] A. A. Hudaib, H. N. Fakhouri *et al.*, "Supernova optimizer: a novel natural inspired metaheuristic," *Modern Applied Science*, vol. 12, no. 1, pp. 32–50, 2018.

[15] A. Kaveh, S. M. Hosseini, and A. Zaerreza, "A physics-based metaheuristic algorithm based on doppler effect phenomenon and mean euclidian distance threshold," *Periodica Polytechnica Civil Engineering*, vol. 66, no. 3, pp. 820–842, 2022.

[16] B. Doğan and T. Ölmez, "A new metaheuristic for numerical function optimization: Vortex search algorithm," *Information sciences*, vol. 293, pp. 125–145, 2015.

[17] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, "The arithmetic optimization algorithm," *Computer methods in applied mechanics and engineering*, vol. 376, p.113609, 2021.

[18] J.-S. Chou, T. T. P. Pham, and C.-C. Ho, "Metaheuristic optimized multi-level classification learning system for engineering management," *Applied Sciences*, vol. 11, no. 12, p. 5533, 2021.

[19] Y. Shi, "Particle swarm optimization," *IEEE connections*, vol. 2, no. 1, pp. 8–13, 2004.

[20] M. Dorigo, M. Birattari and T. Stutzle, "Ant colony optimization," in *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28-39, Nov. 2006.

[21] D. Karaboga, "Artificial bee colony algorithm," *scholarpedia*, vol. 5, no. 3, p. 6915, 2010.

[22] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in engineering software*, vol. 95, pp. 51–67, 2016.

[23] D. P. Acharjya and A. P. Kauser, "Swarm intelligence in solving bio-inspired computing problems: Reviews, perspectives, and challenges," *Handbook of Research on Swarm Intelligence in Engineering*, pp. 74–98, 2015.

[24] D. P. Heyman and M. J. Sobel, *Stochastic models in operations research: stochastic optimization*. Courier Corporation, 2004, vol. 2.

[25] K.-L. Du, M. Swamy *et al.*, "Search and optimization by metaheuristics," *Techniques and Algorithms Inspired by Nature*, pp.1–10, 2016.

[26] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in engineering software*, vol. 95, pp. 51–67, 2016.

[27] S. Mahmood, N. Z. Bawany, and M. R. Tanweer, "A comprehensive survey of whale optimization algorithm: modifications and classification," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 29, no. 2, p. 899, 2023.

[28] H. M. Mohammed, S. U. Umar, T. A. Rashid *et al.*, "A systematic and meta-analysis survey of

whale optimization algorithm," *Computational intelligence and neuroscience*, vol. 2019, 2019.

[29] P. P. Jadhav and S. D. Joshi, "Woadf: whale optimization integrated adaptive dragonfly algorithm enabled with the tdd properties for model transformation," *International Journal of Computational Intelligence and Applications*, vol. 18, no. 04, p. 1950026, 2019.

[30] M. H. Nadimi-Shahraki, H. Zamani, Z. Asghari Varzaneh, and S. Mirjalili, "A systematic review of the whale optimization algorithm: Theoretical foundation, improvements, and hybridizations," *Archives of Computational Methods in Engineering*, pp. 1–47, 2023.

[31] S. Mahdavi, S. Rahnamayan, and K. Deb, "Opposition based learning: A literature review," *Swarm and evolutionary computation*, vol. 39, pp. 1–23, 2018.

[32] M. Jiaojiao, L. Gulyás, and J. Botzheim, "Comparing lamarckian and baldwinian approaches in memetic optimization," in *International Conference on Computational Collective Intelligence*. Springer, 2023, pp. 521–533.

[33] Y. Sun, T. Yang, and Z. Liu, "A whale optimization algorithm based on quadratic interpolation for high-dimensional global optimization problems," *Applied Soft Computing*, vol. 85, p. 105744, 2019.

[34] P. Du, W. Cheng, N. Liu, H. Zhang, and J. Lu, "A modified whale optimization algorithm with single-dimensional swimming for global optimization problems," *Symmetry*, vol. 12, no. 11, p.1892, 2020.

[35] K. Deep and M. Thakur, "A new crossover operator for real coded genetic algorithms," *Applied mathematics and computation*, vol. 188, no. 1, pp. 895–911, 2007.

[36] S. Mirjalili, "Sca: a sine cosine algorithm for solving optimization problems," *Knowledge-based systems*, vol. 96, pp. 120–133, 2016.

[37] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in engineering software*, vol. 69, pp. 46–61, 2014.

[38] S. M. Almufti, A. A. Shaban, Z. A. Ali, R. I. Ali, and J. A. D. Fuente, "Overview of metaheuristic algorithms," *Polaris Global Journal of Scholarly Research and Trends*, vol. 2, no. 2, pp. 10–32, 2023.

[39] H. Ding, Z. Wu, and L. Zhao, "Whale optimization algorithm based on nonlinear convergence factor and chaotic inertial weight," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 24, p. e5949, 2020.

[40] H. Li, P. Zou, Z. Huang, C. Zeng and X. Liu, "Multimodal optimization using whale optimization algorithm enhanced with local search and niching technique," *Mathematical Biosciences and Engineering*, vol. 17, no. 1, pp. 1–27, 2020.

[41] Y. Guo, H. Shen, L. Chen, Y. Liu and Z. Kang, "Improved whale optimization algorithm based on random hopping update and random control parameter," *Journal of Intelligent & Fuzzy Systems*, no. 1, pp. 363-379, 2021.

[42] R. Salgotra, U. Singh and S. Saha, "On some improved versions of whale optimization algorithm," *Arabian Journal for Science and Engineering*, vol. 44, no. 11, pp. 9653–9691, 2019.

[43] S. Rahnamayan, H. R. Tizhoosh, and M. M. Salama, "Oppositionbased differential evolution," *IEEE Transactions on Evolutionary computation*, vol. 12, no. 1, pp. 64–79, 2008.

[44] M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM computing surveys (CSUR)*, vol. 45, no. 3, pp. 1–33, 2013.

[45] Q. Fan, Z. Chen, Z. Li, Z. Xia, J. Yu and D. Wang, "A new improved whale optimization algorithm with joint search mechanisms for high-dimensional global optimization problems," *Engineering with Computers*, pp. 1–28, 2020.

[46] M. P. K. Reddy and M. R. Babu, "Implementing self adaptiveness in whale optimization for cluster head section in internet of things," *Cluster Computing*, vol. 22, no. 1, pp. 1361–1372, 2019.

[47] Q. Zhang and L. Liu, "Whale optimization algorithm based on lamarckian learning for global optimization problems," *IEEE Access*, vol. 7, pp. 36 642–36 666, 2019.

[48] E. Boesiger, "Evolutionary theories after lamarck and darwin," in *Studies in the Philosophy of Biology*. Springer, 1974, pp. 21–44.

[49] R. Deepa and R. Venkataraman, "Enhancing whale optimization algorithm with levy flight for coverage optimization in wireless sensor networks," *Computers & Electrical Engineering*, vol. 94, p. 107359, 2021.

[50] S. Dadashzadeh, M. Aghaie and A. Zolfaghari, "Optimal design of separation cascades using the whale optimization algorithm," *Annals of Nuclear Energy*, vol. 172, p. 109020, 2022.

[51] B. Alatas, E. Akin and A. B. Ozer, "Chaos embedded particle swarm optimization algorithms," *Chaos, Solitons & Fractals*, vol. 40, no. 4, pp. 1715–1734, 2009.

[52] L. dos Santos Coelho and V. C. Mariani, "Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization," *Expert Systems with Applications*, vol. 34, no. 3, pp. 1905–1913, 2008.

[53] P. Priyanga, V. V. Pattankar and S. Sridevi, "A hybrid recurrent neural network-logistic chaos-based whale optimization framework for heart disease prediction with electronic health records," *Computational Intelligence*, vol. 37, no. 1, pp. 315–343, 2021.

[54] Z. GARİP, M. E. CİMEN, D. KARAYEL,

and A. F. BOZ, "The chaos-based whale optimization algorithms global optimization," *Chaos Theory and Applications*, vol. 1, no. 1, pp. 51–63, 2019.

[55] Z. Hua and Y. Zhou, "Exponential chaotic model for generating robust chaos," *IEEE transactions on systems, man, and cybernetics: systems*, vol. 51, no. 6, pp. 3713–3724, 2019.

[56] L. dos Santos Coelho and V. C. Mariani, "Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization," *Expert Systems with Applications*, vol. 34, no. 3, pp. 1905–1913, 2008.

[57] H. Lu, X. Wang, Z. Fei and M. Qiu, "The effects of using chaotic map on improving the performance of multiobjective evolutionary algorithms," *Mathematical Problems in Engineering*, vol. 2014, no. 1, p. 924652, 2014.

[58] F. H. Zhou and Z. Z. Liao, "A particle swarm optimization algorithm," *Applied Mechanics and Materials*, vol. 303, pp. 1369–1372, 2013.

[59] P. Sharma and S. Raju, "Metaheuristic optimization algorithms: A comprehensive overview and classification of benchmark test functions," *Soft Computing*, vol. 28, no. 4, pp. 3123–3186, 2024.

Vaibhav Godbole is pursuing his PhD from Bharti Vidyapeeth (Deemed to be University), College of Engineering, Pune. Prof. Vaibhav Godbole holds a Master's Degree in Electronics & Telecommunication Engineering from Sardar Patel Institute of Technology, Mumbai, and a Bachelor's Degree in Electrical Engineering from Sardar Patel College of Engineering, Mumbai. He is currently working as an Assistant Professor at Fr. Conceicao Rodigues College of Engineering, Mumbai. He has reviewed many research papers for prestigious journals such as IET Networks, IEEE Wireless Sensor Networks, IEEE Communication Letters, and many more. His research interests include mobile and wireless communications, evolutionary algorithms, genetic algorithms, swarm optimization algorithms for ad hoc networks, and wireless sensor networks. Prof. Godbole is both a student member of IEEE and a professional member of ISTE. Prof. Godbole published a design patent on surveillance device using blockchain technology, in the year 2023. Prof. Godbole received two awards for the guided projects as "best hardware project" in the e-yantra innovative challenge competition organized by IIT, Bombay.

Shilpa Gaikwad obtained her Ph.D. degree in Electronics & Telecommunication from Savitribai Phule Pune University, Pune, in 2017. She has received an M.E. degree from the Bharati Vidyapeeth Deemed University College of Engineering at Bharati Vidyapeeth Deemed University, India, in 2006, and a B.E. degree in Electronics Engineering from Shivaji University, Kolhapur, India, in 1999. She has more than 20 years of teaching experience. She is working as an associate professor in the Department of Electronics Engineering at Bharati Vidyapeeth Deemed University College of Engineering, Pune. She has published more than 50 research papers in a variety of international and national journals and conferences, as well as two patents. She is interested in developing and studying new techniques in communication and networks.