Journal of Techniques, ISSN: 2708-8383, Vol. 7, No. 4, 2025, Pages 1-16
DOI: https://doi.org/10.51173/jt.v7i4.2757

JOURNAL OF TECHNIQUES

Journal homepage: http://journal.mtu.edu.iq

RESEARCH ARTICLE - ENGINEERING (MISCELLANEOUS)

Performance Improvement of Smell Agent Optimization Algorithm Using Chaotic Map

Jeremiah Ifi[1], Ahmed Tijani Salawudeen[1, 2], Bashir Olaniyi Sadiq[1,3], Abubakar Umar[1*]

[1]Department of Computer Engineering, Ahmadu Bello University, Zaria, Kaduna State, Nigeria

[2]Department of Electrical and Electronic Engineering, University of Jos, Jos, Plateau State, Nigeria

[3]Department of Electrical, Telecommunications and Computer Engineering, Faculty of Engineering and Applied Sciences, Kampala International University – Western Campus, Ishaka, Bushenyi District, Uganda

[*] Corresponding author E-mail: abubakaru061010@gmail.com

Article Info.

Article history:

Received
24 September 2025

Revised
09 December 2025

Accepted
18 December 2025

Published
31 December 2025

Abstract

Metaheuristic algorithms have become dominant in solving different kinds of optimization problems due to their simplicity, adaptability and derivative-free approach. The Smell Agent Optimization (SAO) algorithm is a recent metaheuristic algorithm that is inspired by the concept of smell perception. The algorithm operates in three modes known as sniffing, trailing and random mode. The sniffing mode was modelled based on how an agent perceives the smell molecules. The trailing mode was modelled based on how an agent trails a smell molecule to identify its source. The random mode is a strategy employed by the algorithm to escape the state of confusion known as the local minimum. The SAO just like other metaheuristic algorithms has the problem of local minima, imbalance between exploration and exploitation and slow convergence as a result of the different modes involved. Chaotic maps have been shown to improve the performance of metaheuristic algorithms. The sinusoidal, logistic and singer maps were introduced in each of the modes of SAO to form a new algorithm known as chaotic smell agent optimization (cSAO). This modification was to improve its general performance and convergence of the original SAO. The cSAO was tested on seventeen benchmark functions and the results obtained were compared with SAO and PSO. The statistical result showed that cSAO and SAO obtained the best solution in 12 functions and PSO in 10 functions but cSAO is ranked higher than SAO and PSO with final rank values of 1.33, 1.66 and 1.86 respectively. The cSAO also converges faster than SAO by 25% but fails with PSO due to the number of function evaluations and high exploitation rate of PSO.

Publisher: Middle Technical University

Keywords: Chaotic Maps; Exploration; Exploitation; Function Evaluation; Local Minima.

1. Introduction

Metaheuristic algorithms have become very famous over the last two decades. Obviously, some of them such as Particle Swarm Optimization (PSO), Genetic Algorithm (GA) and Ant Colony Optimization (ACO) have become very popular in the field of engineering and science in solving optimization problems. These algorithms are simple, adaptable, and derivation-free methods [1]. Metaheuristic algorithms behave stochastically; they start the optimization process by producing random solutions, as opposed to gradient search approaches, which require the search space derivative to be calculated. The exceptional ability of metaheuristic algorithms to avoid the algorithm's premature convergence is their defining feature. The stochastic behavior of algorithms allows the strategies to operate as a "black box," avoid local optimum, and effectively and efficiently explore the search space. The algorithms trade-off between exploration and exploitation, are the two most crucial and fundamental features [2, 3]. During the exploration phase, the algorithms thoroughly examine the promising search space, while the exploitation phase involves a local search of any promising areas that were discovered.

Based on their behavior, Metaheuristic algorithms are further categorized into four groups: evolution-based, physics-based, human-related and swarm intelligence-based algorithms [4]. Fig. 1 shows the categorization of the algorithms [5].

Evolution-based algorithms are algorithms that are inspired by the natural phenomena of evolution. It begins its operation by randomly generating a population of solutions. The best solution is used to create a new individual solution using mutation, crossover and selection. Examples of evolution-based algorithms are Genetic Algorithm (GA) based on Darwin's evolution method, Evolution Strategy (ES), Genetic Programming (GP), Tabu search and Differential Evolution (DE)[4] .

Physics-based algorithms are algorithms inspired by the rules of physics in the universe. They also start by random generation of the initial population which continues until the best solution is obtained. Examples of such algorithms are Simulated Annealing (SA) and Harmony Search [6].

Nomenclature & Symbols			
SAO	Smell Agent Optimization	cSAO	Chaotic Smell Agent Optimization
PSO	Particle Swarm Optimization	GA	Genetic Algorithm
ACO	Ant Colony Optimization	ES	Evolution Strategy
GP	Genetic Programming	DE	Differential Evolution
SA	Simulated Annealing	TLBO	Teaching Learning-Based Algorithm
LCA	League Championship Algorithm	ACO	Ant Colony Optimization
HBSO	Honey Bee Swarm Optimization	MO	Monkey Optimization

Human behavior-related algorithms are techniques purely inspired by human behavior. Every human being has a special and distinct way of doing any activities that affects their performance. This concept motivated researchers to develop algorithms that help solve real-life problems. The popular example in this class is what is known as the Teaching Learning-based Algorithm (TLBO) and the League Championship Algorithm (LCA) [4].

Swarm intelligence algorithms are inspired, among other things, by the social behaviors of insects, animals, fish, and birds. Kennedy and Eberhart invented the well-known PSO technique. It is inspired by the behavior of a flock of birds flying through the search space in search of the best location (position). Other examples of swarm intelligence algorithms include Ant Colony Optimization (ACO), Honey Bee Swarm Optimization (HBSO), and Monkey Optimization (MO) [5].

Smell agent optimization algorithm (SAO) is a recently developed metaheuristic algorithm inspired by the sense of smell and trailing behavior of an agent such as humans, dogs, sharks, insects and bacteria to identify the smell source [7]. The SAO has been successfully utilized in solving different kinds of problems such as Path Planning [8], Capacitated Vehicle Routing [9], Frequency Stabilization in Micro-grid [10], Hybrid Renewable Energy System [7], and Coordination Scheme for a Distribution Network [11]. However just like any other metaheuristic algorithm, it faces the problem of imbalance between exploration and exploitation, local minima trapping and slow convergence.

The outline of the article is as follows; Section one presents the introduction while Section two gives some background on the review of metaheuristic algorithms. Section three elaborates on the development of the chaotic smell agent optimization algorithm (cSAO). Section four is the results and discussion, and finally the conclusion and recommendations are given in section five

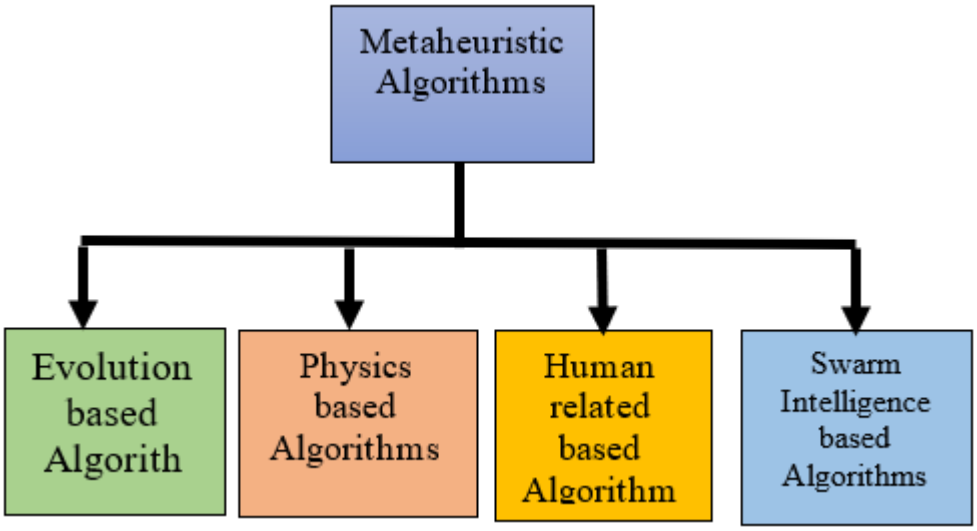

Fig. 1. Categorization of Metaheuristic Algorithms [5]

2. Review of Metaheuristic Algorithm

This section focuses on the review of the metaheuristic algorithms such as particle swarm optimization and the smell agent optimization algorithms as well as the chaotic maps utilized in this work.

2.1. Particle swarm optimization

Particle swarm optimization (PSO) is a swarm intelligence-based method developed by Kennedy and Eberhart in 1995 inspired by the natural behavior of schooling of fish and the flocking of birds in search for food [12]. The populations are randomly initialized in particle swarm optimization, just like in evolutionary algorithms like the genetic algorithm (GA). Every particle can recall its own personal best place in its learning, denoted by the letters pbest, and its best position within the present population, denoted by the letters gbest. Every particle moves at a certain pace in the direction of these two locations, always shifting its position. The following two equations were used to update the velocity and position of the particle [13].

$$v_i^{(t+1)} = w \times v_i^t + c_1 \times r_1 \left(p_{best}^{(t)} - x_i^t\right) + c_2 \times \qquad r_2(g_{best} - x_i^{(t)} \tag{1}$$

$$x_i^{(t+1)} = v_i^{(t+1)} + x_i^{(t)} \tag{2}$$

where w is the controlling parameter [13]. The cognitive component c1 and the social component c2 are thought to be learning factors that direct searches for the individual best and tShe global best of particles in the process of evolution [14], r1 and r2 are two uniformly generated random numbers between 0 and 1, and t represents the number of iterations. The flow chart for the PSO algorithm is shown in Fig. 2. [14]

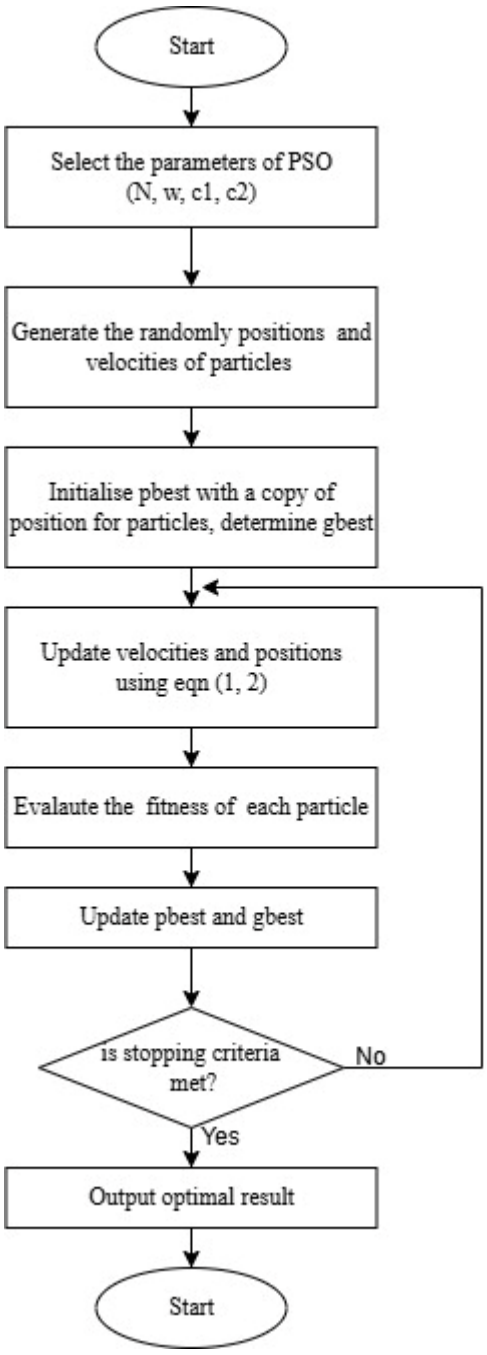

Fig. 2. The Flow Chart of the Particle Swarm Optimization

2.2. Smell Agent Optimization (SAO) algorithm

The smell agent optimization algorithm is an optimization algorithm that was inspired by the sense of smell and the ability of an agent to discriminate and trail the smell source [8, 15]. The algorithm works in three distinct modes: sniffing, trailing, and random mode. Each of these modes is discussed as follows:

2.2.1. Sniffing mode

The sniffing mode serves as the SAO's starting point. The smell molecules evaporate in the direction of the agent. The agent sniffs the smell molecules and determines which molecule has the highest fitness. This is considered the location of an agent. In implementing the algorithm, the location of these molecules is obtained by randomly generating the set of numbers using [15]:

$$X_i^t = LB + r_1 \times (UB - LB) \quad (3)$$

Where LB and UB represent the lower and upper bounds defined for the decision variable, r1 is a random number generator in the range of (0,1). Xti is the position of smell molecules generated with population (N) and dimension (D) represented as [7]:

$$\text{Xti}=\begin{bmatrix} x_{(1,1)} & \cdots & x_{(1,D)} \\ \vdots & \ddots & \vdots \\ x_{(N,1)} & \cdots & x_{(N,D)} \end{bmatrix} \quad (4)$$

From equation (2), since the smell molecules are moving in the direction of the agent, they move with a uniform velocity. Therefore, the velocity of the smell molecule can also be represented as [7]:

$$V_i^t = \begin{bmatrix} v_{(1,1)} & \cdots & v_{(1,D)} \\ \vdots & \ddots & \vdots \\ v_{(N,1)} & \cdots & v_{(N,D)} \end{bmatrix} \quad (5)$$

As the agent moves in the solution search space the velocity and position of smell molecules are updated with the following equations [15]:

$$V_I^{t+1} = V_i^t + r_2 \times \left(\frac{3kT}{m}\right) \quad (6)$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \quad (7)$$

Where V_I^{t+1} is the updated velocity, V_i^t is the previous velocity, k is the Boltzmann constant, T is the absolute temperature, m is the mass of the smell molecules and r_2 is a random number generator that penalizes the velocity update. $X_i^{t+1}$and X_i^t represent the updated position and previous position of the smell molecules respectively [15].

2.2.2. Trailing mode

The next stage of SAO is the trailing mode. In this mode the agent trails the molecule based on the concentration of the smell molecules. It is generally believed that as the agent approaches the smell source, the concentration/intensity of the smell molecules increases. Thus, the concentration of the smell molecule is critical during trailing. Furthermore, an agent's olfactory capacity should be strong to aid the process. During trailing, the agent observes the molecules with the best and worst fitness, allowing the agent to follow an optimal path while trailing. The mathematical equation governing the trailing mode is as follows [7, 15]:

$$X_i^{t+1} = X_i^t + r_3 \times ofc \times \left(X_{Agent}^t - X_i^t\right) - r_4 \times \qquad ofc \times (X_{worst}^t - X_i^t) \quad (8)$$

Where ofc is the olfaction capacity of an agent, r_3 and r_4 are random number generators that penalize the sniffing mode.

2.2.3. Random mode

The random mode on the other hand occurs when an agent is confused or trapped in the local minimum during trailing. This mode helps the agent to assume any random step in the search space to continue trailing or the agent goes back and starts the whole process again. The mathematical expression is represented as: [7, 15].

$$X_i^{t+1} = X_i^t + r_5 \times \theta \quad (9)$$

Where θ represents the step movement and r_5 is a random number generator that penalizes the random m]ode.

2.3. Chaos theory

Chaos is an unusual form of deterministic randomness that appears in nonlinear dynamical systems that are non-periodic, non-converging, and bounded [16]. Chaos exhibits properties such as determinism, non-linearity and sensitivity dependence [17]. Deterministic suggests that a discrete or continuous mathematical equation can adequately capture the behavior of a chaotic system. Non-linearity in the sense that there is no direct relationship between the input and output. Sensitivity to initial conditions such as the butterfly effect which states that the flapping of the wings of a butterfly in Brazil can cause a Tornado in another part of the world such as Texas. This implies that a small difference in initial condition such as an error due to measurement or rounding off numbers can cause a significant variation in the dynamics of the system therefore rendering the system unpredictable in the long run. These illustrate sensitivity dependence on initial condition [18].

2.3.1 Chaotic map

Different chaotic maps with various mathematical functions are used to incorporate the concept of chaos theory into an optimization algorithm. These chaotic maps use some mathematical functions to generate chaotic variables rather than random variables which perform searching at high speed compared to the random variables. Some examples of these chaotic maps reported in the literature in the field of optimization are discussed as follows [19].

- Sinusoidal Map: The sinusoidal map is expressed mathematically as [20]:

$$x_{n+1} = ax_n^2 \sin(\pi x_n) \quad (10)$$

Where xn+1 is the sinusoidal chaotic number to be generated, xn is the starting position of the map, $and\ a$ is a control parameter which is usually assumed as $a = 2.3$.

- Logistic Map: The logistic map is a famous one-dimensional chaotic map which can generate a chaotic variable in the range of [0,1] [20]. The mathematical equation is described by:

$$x_{n+1} = k(1 - x_n) \quad (11)$$

Where: xn+1 is the logistic map generator, xn is the starting position of the map, and r is a control parameter which is usually taken as $k = 4$ for a chaotic system.

- Singer Map: The singer map is defined by the mathematical equation as [20]:

$$x_{n+1} = a(7.86x_n - 23.32x_n^2 + 28.75x_n^3 - 13.302875x_n^4) \quad (12)$$

Where xn+1 is the chaotic singer map generator, xn is the starting position of the map, a is a control parameter which has values as $a = 1.07$.

3. Development of a Chaotic Smell Agent Optimization Algorithm (cSAO)

This section explains in detail the development of the chaotic SAO as well as the benchmark function used to validate any of the optimization functions.

3.1. Chaotic Smell Agent Optimization Algorithm (CSAO)

The CSAO incorporates the concept of chaotic maps into each of the modes of the original SAO to improve the exploitation capability of the algorithm as well while not undermining its exploration ability. Since SAO operates in three modes, three chaotic maps are also employed in each of the modes to guide the search process and enhance its overall performance as well. The detailed mathematical expression of each of the modified modes using the chaotic map is shown in the subsequent sub-section.

3.1.1. Chaotic sniffing mode

From the sniffing mode shown in equation (6), the parameter r_1 is an important parameter that penalizes the sniffing mode. In the original SAO this was obtained using a standard probability distribution. In the chaotic SAO this can be generated using the sinusoidal chaotic map described by the mathematical expression in equation (10). If *n=0*, the starting position becomes x_0 and the sinusoidal chaotic map becomes x_1 which simplifies the equation as:

$$x_1 = ax_o \sin(\pi x_o) \quad (13)$$

Equation (13) is transformed into a function as x_1 = S*inusoidal(r)* where $r = x_0$. Therefore, the chaotic sniffing mode can be expressed as

$$X_i^{t+1} = X_i^t + V_i^{t+1} + Sinusoidal(r) \times \left(\frac{3kT}{m}\right) \quad (14)$$

where *sinusoidal(r)* is a random number generated by a sinusoidal chaotic map and r is any arbitrary random number in the range of 0 and 1.

3.1.2. Chaotic trailing mode

Similarly, from equation (6) the random numbers r_3 and r_4 were generated using the logistic map represented by c_1 and c_2 described in equation (15)

$$c_{n+1} = k(1 - c_n) \quad (15)$$

When n=1, equation (15) becomes

$$c_2 = k(1 - c_1) \quad (16)$$

c_1 was generated using a random number that is in the range of 0 and 1 while c_2 was also generated using equation (3.4) with $k = 4$ for a chaotic system.

Therefore, the chaotic trailing mode is expressed mathematically as:

$$X_i^{t+1} = X_i^t + c_1 \times ofc \times \left(X_{Agent}^t - X_i^t\right) - c_2 \times ofc \times (X_{worst}^t - X_i^t) \quad (17)$$

This modification is aimed at increasing the searching speed of the mode and subsequently converges to an optimal solution in less time.

3.1.3. Chaotic random mode

The random mode equation is described using equation (7), the parameter r_5 is also generated chaotically using the Singer map described by equation (18). Therefore, the chaotic random mode equation is expressed as:

$$X_i^{t+1} = X_i^t + singer(r) \times \theta \quad (18)$$

Where singer (r) is a singer chaotic map and r is a random number generator in the range of 0 to 1. This modification is to improve the level of randomness in the mode. The flowchart of the chaotic smell agent optimization algorithm is shown in Fig. 3.

3.2. Optimization benchmark functions

Every new or modified optimization algorithm is typically evaluated using a standard set of applied mathematical functions. These functions are either unimodal or multimodal. Unimodal functions have a single global optimum which helps to examine the exploitation ability of the algorithm while multimodal functions have two or more global optima which are used to examine the exploration capability of the algorithm. In this article, a total of seventeen (17) functions with different properties were used to evaluate the performance of the cSAO. These functions are shown in Table 1. Enhance the exploitation ability that will enable an agent to converge in a shorter time.

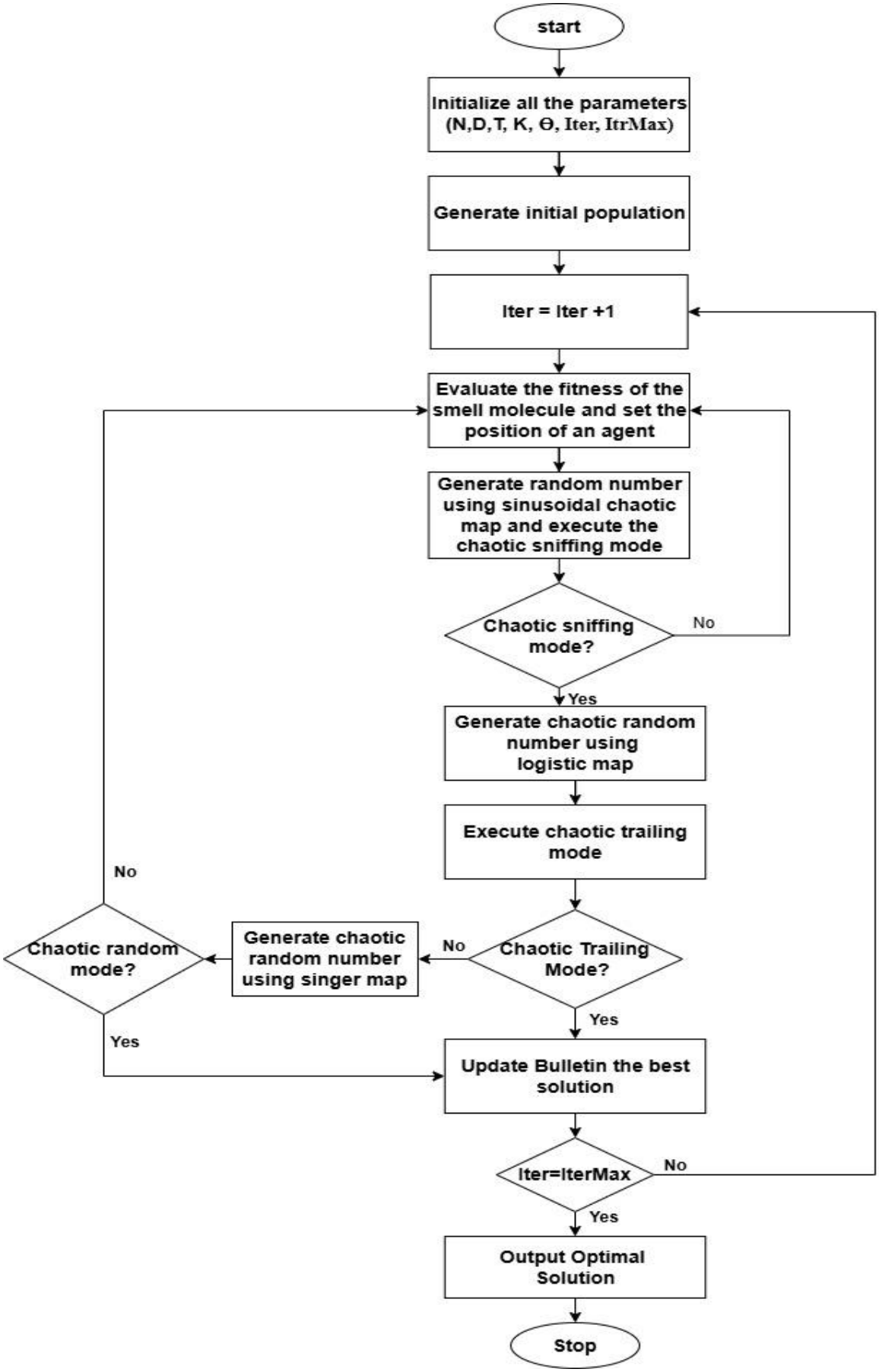

Fig. 3. Flowchart of the Chaotic Smell Agent Optimization Algorithm

Table 1. Optimization benchmark functions [7]

Fn	Name	Dimension	Description	Fmin
1	Bird	2	$f(x_1,x_2)=\sin(x_1)e^{(1-\cos(x_2))^2}+(x_1-x_2)^2$	-106.765
2	Bohachevsky1	2	$f(x_1,x_2)=x_1^2+2x_2^2-0.3\cos(3\pi x_1)-0.4\cos(4\pi x_2)+0.7$	0
3	Camel-six Hump	2	$f(x_1,x_2)=(4-2.1x_1^2+\frac{x_1^4}{3})x_1^2+x_1x_2+(-4+4x_2^2)x_2^2$	-1.0316
4	Chichinadze	2	$f(x_1,x_2)=x_1^2-12x_1+11+10\cos(0.5\pi x_1)$ $+8\sin(2.5\pi x_1)-(0.2)^{0.5}\exp(-0.5(x_2-0.5)^2)$	-42.944
5	Matyas	2	$f(x_1,x_2)=0.26(x_1^2+x_2^2)-0.48x_1x_2$	0
6	Michalewicz (	2	$f(x)=-\sum_{i=1}^{d}\sin(x_i)\sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$	-1.8013
7	Quadratic	2	$f(x)=-3803.84-138.08x_1-232.92x_2$ $+128.0x_1^2+203.64x_2^2+182.25x_1x_2$	-3873.72
8	Schaffer function	2	$f(x)=\sum_{i=1}^{30}\left(x_i^2+x_{i+1}^2\right)^{0.5}\left\{\left[\sin 50\left(x_i^2+x_{i+1}^2\right)^{0.1}\right]^2\right\}$	0
9	Styblinki's Tang	2	$f(x)=\sum_{i=1}^{n}(x_i^4-16x_i^2+5x_i)$	-78.332
10	Box-Betts Function	3	$f(x_1,x_2)=\sum_{i=1}^{k}(e^{-0.1(i+1)x_1}-e^{-0.1(i+1)x_2}-[(e^{-0.1(i+1)})-e^{-(i+1)x_3})])^2$	0
11	Colville	4	$f(x)=\sum_{i=1}^{D}x_i^6(2+\sin(\frac{1}{x_i})$	0
12	Michalewicz	5	$f(x)=-\sum_{i=1}^{d}\sin(x_i)\sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$	-4.6877
13	Michalewicz2	10	$f(x)=-\sum_{i=1}^{d}\sin(x_i)\sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$	-9.6602
14	Shubert	5	$f(\mathrm{X})=\left(\sum_{i=1}^{n}i\cos((i+1)x_i+i)\right)\left(\sum_{i=1}^{n}i\cos((i+1)x_{i+1}+i)\right)$	-186.73
15	Ackley	50	$f(X)=-2o\exp\left[-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right]-\exp\left[\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right]+20+e$	0
16	Sphere	30	$f(x)=\sum_{i=1}^{d}x_i^2$	0
17	Rosenbrock	30	$f(x)=\sum_{i=1}^{n-1}((x_i-1)^2+100(x_{i+1}-x_i^2)^2)$	0

In addition to cSAO, other metaheuristic algorithms such as SAO, PSO and AOA were also evaluated on the benchmark functions for performance comparison. All the algorithms were simulated using MATLAB R2016a and implemented on an HP computer system running Windows 10 Pro with Intel Core i5 Gen CPU @ 1.9GHZ and 6.0 GHz RAM. A population of 50 was used for all the algorithms to reduce the effect of parameters on the performance of the algorithms. Other parameters are summarized in Table 2 [7].

Table 2. Parameter setting of algorithms [8]

SN	Algorithms	Parameters
1	cSAO	m=2.4, T=0.95,K=0.6,*ofc=0.9*,$\theta = 0.02$
2	SAO	m=2.4, T=0.95,K=0.6,*ofc=0.9*,$\theta = 0.02$
3	PSO	C1=1.5, C2=2.0, w=1

Where; m=mass of smell molecules, T=Temperature of smell molecules, K=Boltzman constant, ofc=Olfaction capacity, θ=Step movement, C1=Cognitive components, C2=Social components, w=Initial weight.

4. Results and Discussion

This subsection presents the experimental results and discussion. It also gives the convergence analysis of the results obtained from the experimental results.

4.1. Experimental result and discussion

For all experimental results carried out on the benchmark functions, the algorithms were run thirty (30) times and in each run, five hundred (500) iterations were used for cSAO and SAO while one thousand (1000) iterations were used for PSO. The best, worst, average, standard deviation, and rank of each algorithm were determined after thirty runs. The Algorithms' performance was ranked based on the best value obtained from the benchmark function. In a scenario where two or more algorithms obtained the best value, the average, standard deviation, and the worst result were then used to determine the best result for the algorithms. For instance, if n algorithms all returned the same best value for the benchmark function, they would all rank the same, and the next algorithm would rank n+1. The average rank on the benchmark functions was computed at the end of each table. The algorithms with the best performance were then identified by calculating the average rank, which also served to determine the final rank of the algorithms. The statistical result obtained for the benchmark functions is shown in Table 3. From the result it is seen that cSAO and SAO obtained the best solution in 12 functions (F1 to F5, F7 to F11, F14 and F15) which constitute 80% of the benchmark function. Although cSAO and SAO obtained the best solution in the same number of functions, cSAO is ranked ahead of SAO in terms of solution to the global optimal.

The PSO obtained the best solution in 10 functions (F2 to F7, F9 to F11, F14 and F15) which constitute 66.67% of the benchmark functions. None of the algorithms obtained the best solution in F6, F12 and F13. However, the result obtained by cSAO appears to be closer to the optimal solution as compared with the SAO and PSO respectively.

The overall ranking of the algorithms also shows that cSAO is ranked first followed by SAO and PSO with the average rank of 1.3333, 1.6667 and 1.8667 respectively. This result showed the superior performance of the cSAO when compared with SAO and PSO.

4.2. Convergence analysis

The convergence analysis of the algorithm is presented in the plots shown in Figs. 4 to 20. From these plots it was observed that cSAO converges faster than the SAO in 9 functions (F1, F3, F4, F5, F6, F7, F10, F12 and F16) out of the 16 selected benchmark functions which constitute 56.25% of the functions. The SAO convergence in 6 functions (F2, F9, F11, F13, F14 and F15) which is 37.25% of the functions. This showed an improvement of 25% of cSAO over SAO. This improvement is expected because of the chaotic behavior introduced in each of the modes for improved search. However, for function F8, cSAO and SAO converged at the same iteration. In comparison of the convergence of cSAO with that of PSO. The PSO algorithm converges in 10 functions (F1, F2, F3, F4, F6, F7, F8, F9, F13 and F15) which is 56.25%. The cSAO converges in 5 (F10, F11, F12, F14 and F16) which is 31.25%. For function F5, cSAO and PSO appeared to converge at the same iteration. This constitutes a decrease in convergence of cSAO over PSO by 25%. This decrease in convergence is attributed to the fact that PSO operates in single mode compared to cSAO which operates in three modes. Secondly, PSO operates in two function evaluations compared to cSAO which operates in four function evaluations. Finally, PSO has a high rate of exploitation over exploration, which makes the algorithm converge to a solution that may not necessarily be the best solution. The cSAO which is a variant of SAO has improved the high rate of exploration of SAO with a chaotic map.

Table 3. Results obtained for the benchmark functions

Fn	Performance Matrics`	cSAO	SAO	PSO	Global Best
	Mean	-106.776	-106.775	-106.787	
	Best	-106.765	-106.765	-106.787	
1	Std	9.32E-03	1.50E-02	7.095E-14	-106.765
	Worst	-106.787	-106.787	-106.787	
	Rank	1	2	3	
	Mean	0.0000	0.0000	0.0000	
	Best	0.0000	0.0000	0.0000	
F2	Std	0.0000	0.0000	0.0000	0
	Worst	0.0000	0.0000	0.0000	
	Rank	1	1	1	
	Mean	-1.0316	-1.0316	-1.0316	
	Best	-1.0316	-1.0316	-1.0316	
F3	Std	0.000	0.000	0.000	-1.0316
	Worst	-1.0316	-1.0316	-1.0316	
	Rank	1	1	1	
	Mean	-42.940	-42.907	-42.825	
	Best	-42.944	-42.944	-42.944	
F4	Std	3.472E-02	3.747E-02	2.014E-01	-42.944
	Worst	-42.788	-42.811	-42.497	
	Rank	1	2	3	
	Mean	0.0000	0.0000	0.0000	
	Best	0.0000	0.0000	0.0000	
F5	Std	0.0000	0.0000	0.0000	0
	Worst	0.0000	0.0000	0.0000	
	Rank	1	1	1	

Continue Table 3. Results obtained for the benchmark functions

	Mean	-1.9999	-1.9999	-2.000	
	Best	*-1.9998*	-1.9999	-2.000	
F6	Std	1.773E-05	2.309E-05	0.000	-1.8013
	Worst	-1.9999	-1.9999	-2.000	
	Rank	1	2	3	
	Mean	-3873.63	-3873.62	-3873.72	
	Best	-3873.72	-3873.72	-3873.72	
F7	Std	1.02E-02	0.101	1.387E-12	-3873.72
	Worst	-3873.59	-3873.39	-3873.72	
	Rank	2	3	1	
	Mean	0.0000	0.0000	5.320E-30	
	Best	0.0000	0.0000	1.397E-30	
F8	Std	0.0000	0.0000	8.730E-30	0
	Worst	0.0000	0.0000	2.719E-29	
	Rank	1	1	4	
	Mean	-78.329	-78.329	-78.332	
	Best	-78.332	-78.332	-78.332	
F9	Std	2.93E-03	2.27E-03	1.445E-14	-78.332
	Worst	-78.320	-78.323	-78.332	
	Rank	3	2	1	
	Mean	0.0000	0.0000	0.0000	
	Best	0.0000	0.0000	0.0000	0
F10	Std	0.0000	0.0000	0.0000	
	Worst	0.0000	0.0000	0.0000	
	Rank	1	1	1	
	mean	0.0000	0.0000	0.0000	
	Best	0.0000	0.0000	0.0000	0
F11	Std	0.0000	0.0000	0.0000	
	Worst	0.0000	0.0000	0.0000	
	Rank	1	1	1	
	mean	-4.999	-4.998	-5.000	
	Best	*-4.999*	*-4.999*	-5.000	
F12	Std	5.35E-05	6.21E-05	0.000	-4.6877
	Worst	-4.999	-5.000	-5.000	
	Rank	1	2	3	
	mean	-9.9999	-9.9999	-10.000	
	Best	*-9.9996*	*-9. 9996*	-10.000	
F13	Std	1.00E-05	8.886E-05	0.000	-9.6602
	Worst	-9.9999	-9.9999	-10.000	
	Rank	1	2	3	
	mean	-4.999	-4.998	-5.000	
	Best	-4.999	-4.999	-5.000	
F14	Std	5.35E-05	6.21E-05	0.000	-4.6877
	Worst	-4.999	-5.000	-5.000	
	Rank	1	2	3	
	mean	8.8817E-16	8.8817E-16	8.8817E-16	
	Best	8.8817E-16	8.8817E-16	8.8817E-16	
F15	Std	0.0000	0.0000	0.0000	0
	Worst	8.8817E-16	8.8817E-16	8.8817E-16	
	Rank	1	1	1	
	mean	0.0000	0.0000	0.0000	
	Best	0.0000	0.0000	0.0000	0
F16	Std	0.0000	0.0000	0.0000	
	Worst	0.0000	0.0000	0.0000	
	Rank	1	1	1	
	mean	7497E-03	7.877E-03	0.0000	
	Best	3.815E-05	6.506E-04	0.0000	
F17	Std	7.857E-03	7877E-03	0.0000	0
	Worst	3.745E-02	2.636E-02	0.0000	
	Rank	2	4	1	
Count of Global Best		12	12	10	
Average Ranking		1.3333	1.6667	1.8667	
Final Rank		1	3	3	

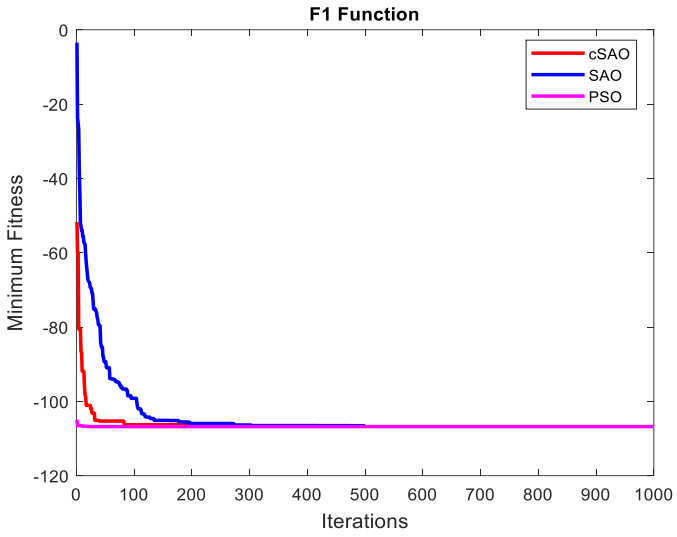

Fig. 4. Convergence plot for bird function

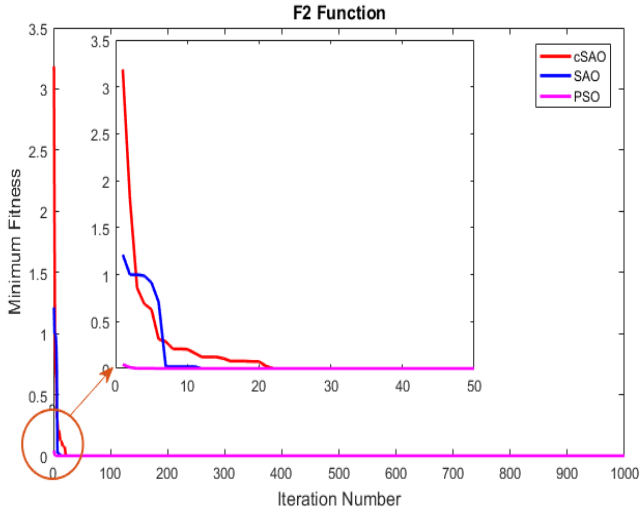

Fig. 5. Convergence plot for bohachevsky1

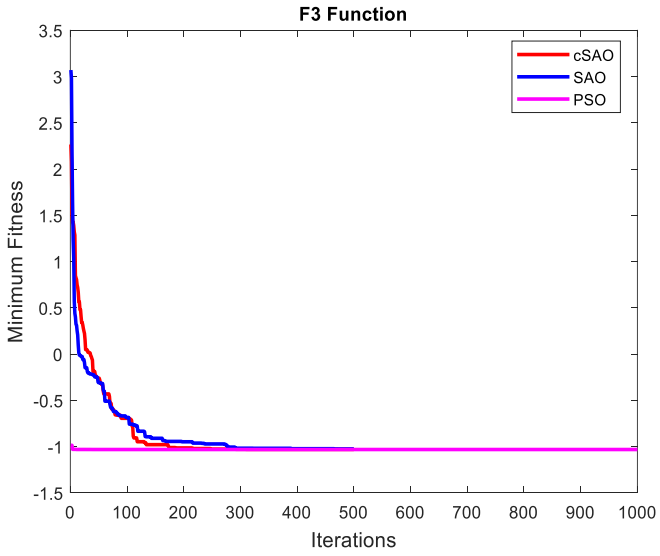

Fig. 6. Convergence plot for camel-six hump function

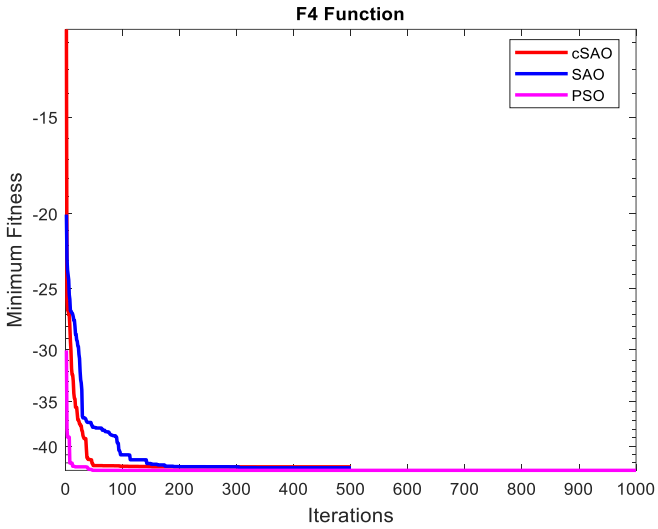

Fig. 7. Convergence plot for chichinadze function

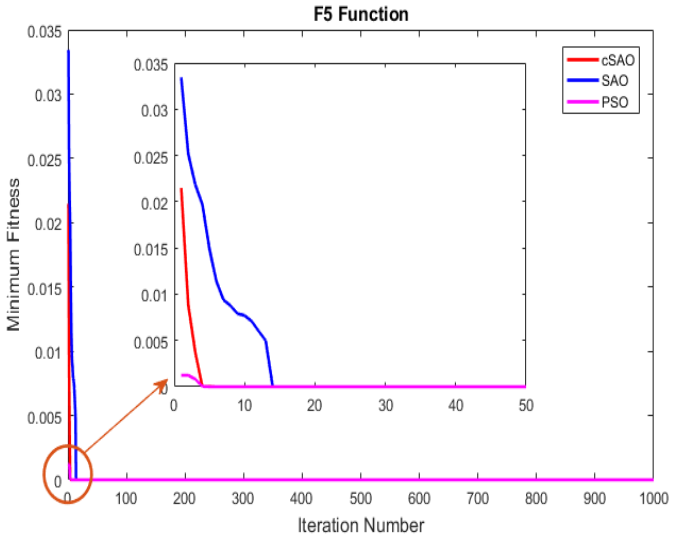

Fig. 8. Convergence plot for matyas function

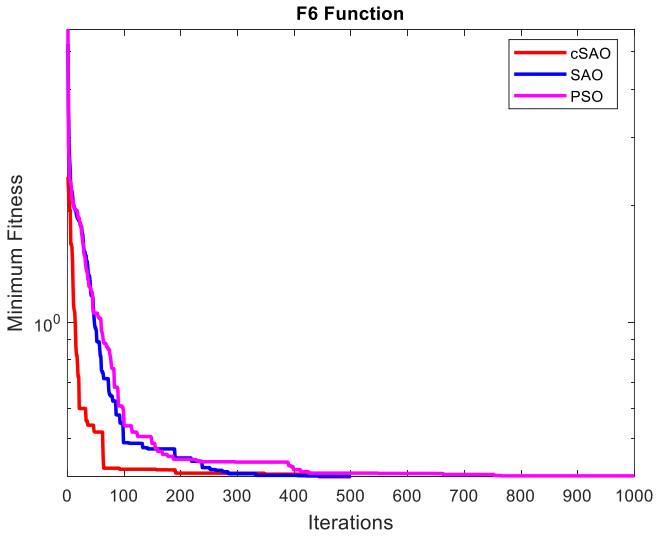

Fig. 9. Convergence plot for michalewicz

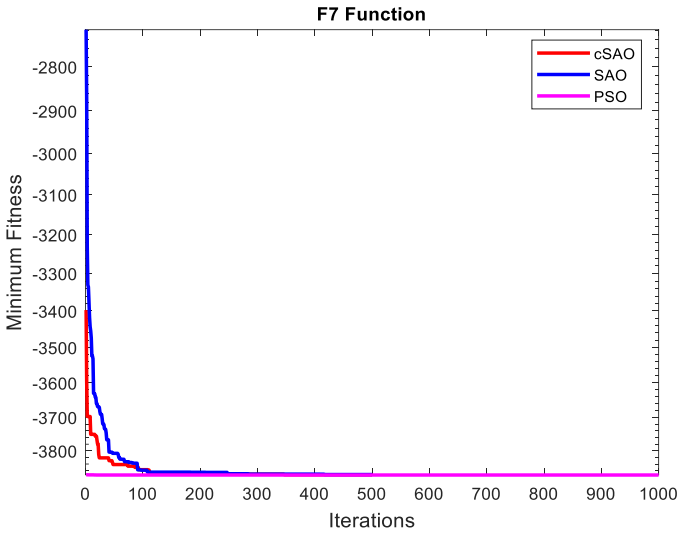

Fig. 10. Convergence plot for quadratic function

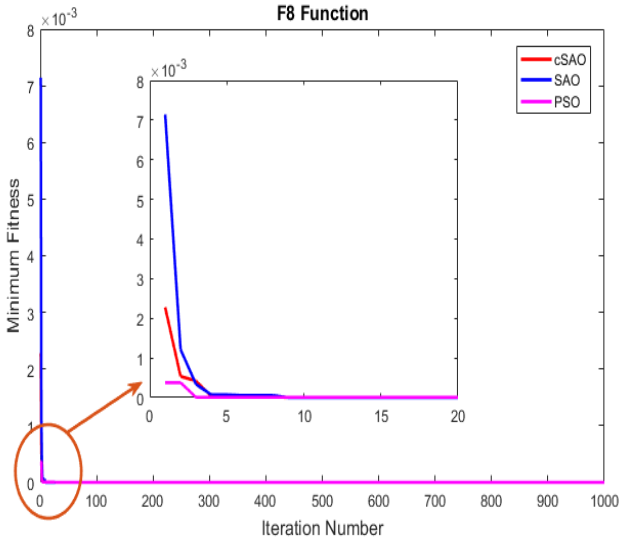

Fig. 11. Convergence plot for Schaffer function

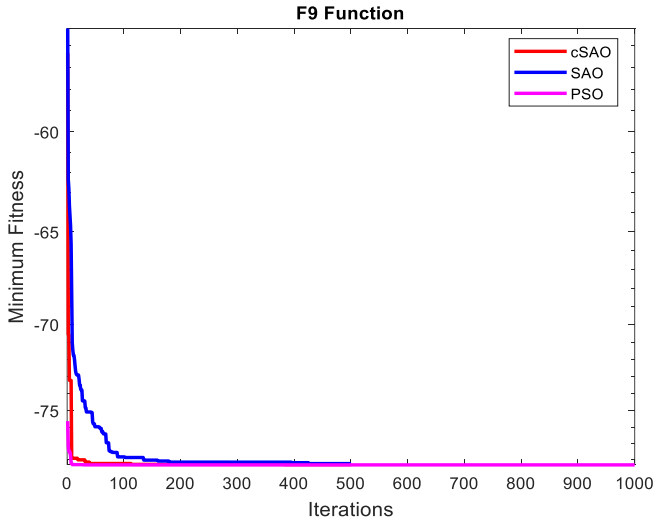

Fig. 12. Convergence plot for Styblinki's tang function

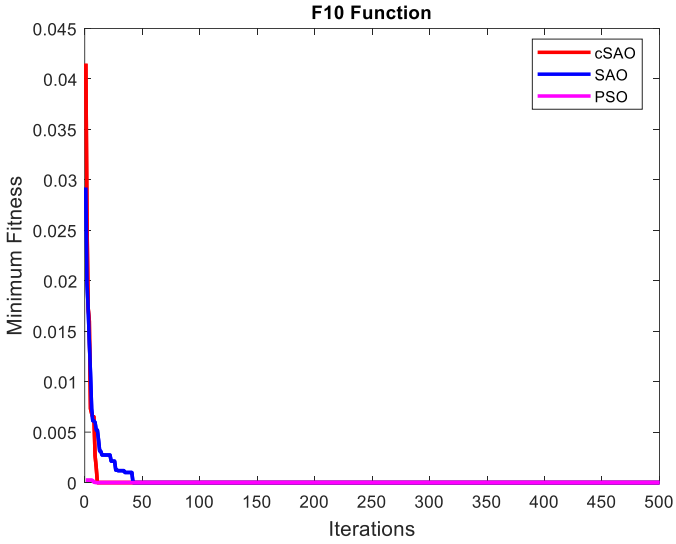

Fig. 13. Convergence plot for box-betts function

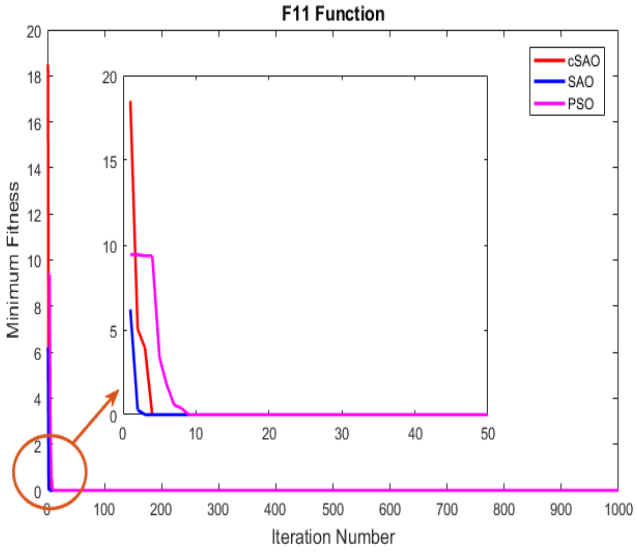

Fig. 14. Convergence plot for Colville function

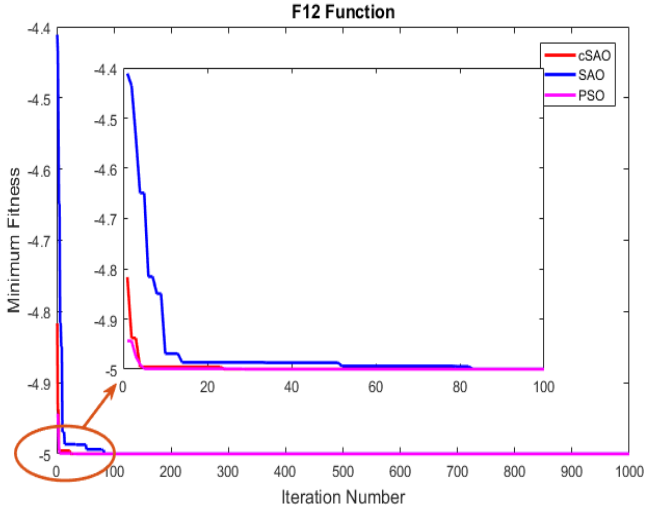

Fig. 15. Convergence plot for Michalewicz function

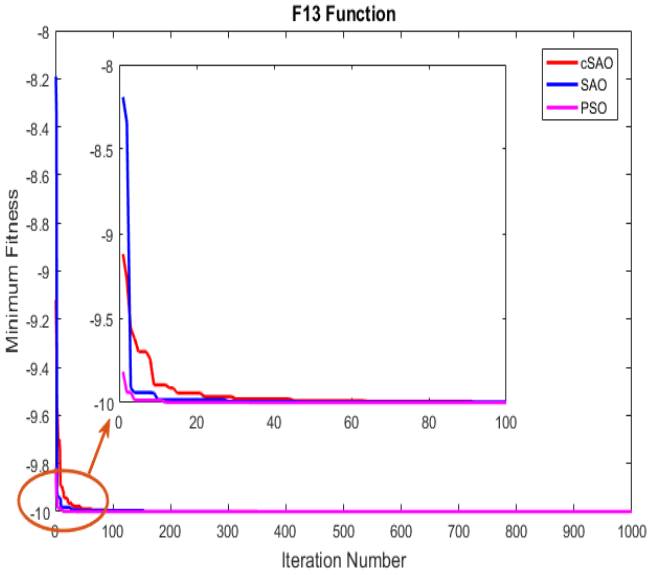

Fig. 16. Convergence plot for Michalewicz2 function

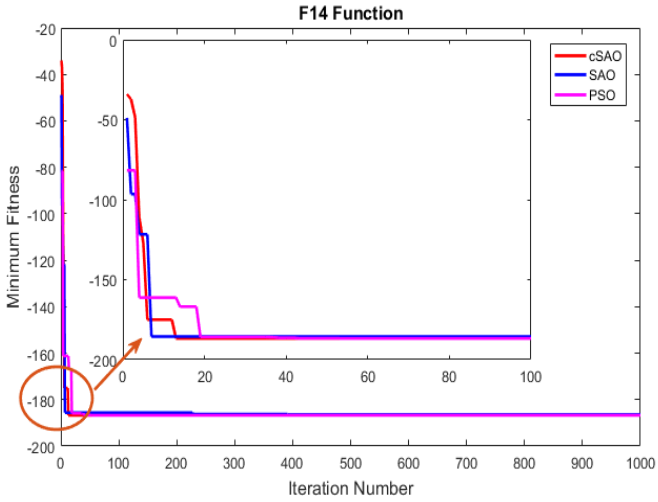

Fig. 17. Convergence plot for Shubert function

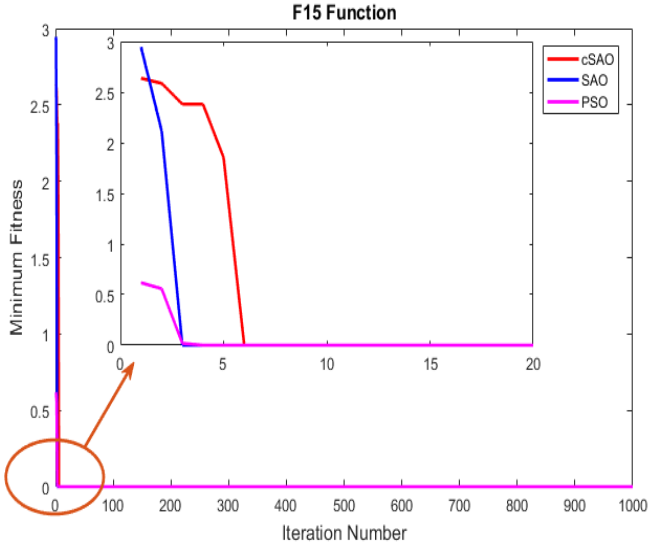

Fig. 18. Convergence plot for Ackley function

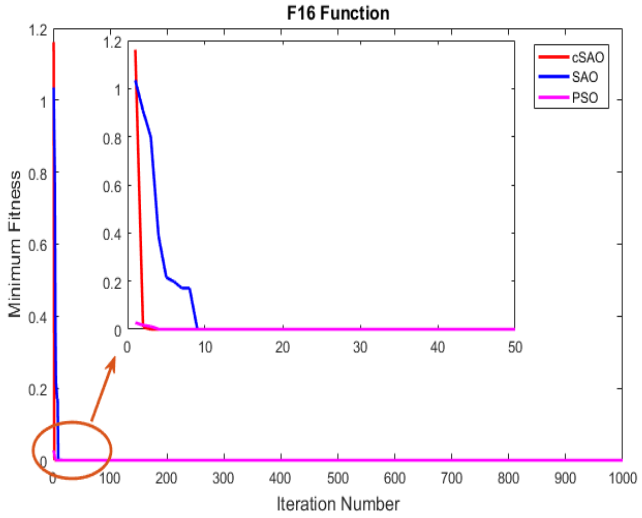

Fig. 19. Convergence plot for Sphere function

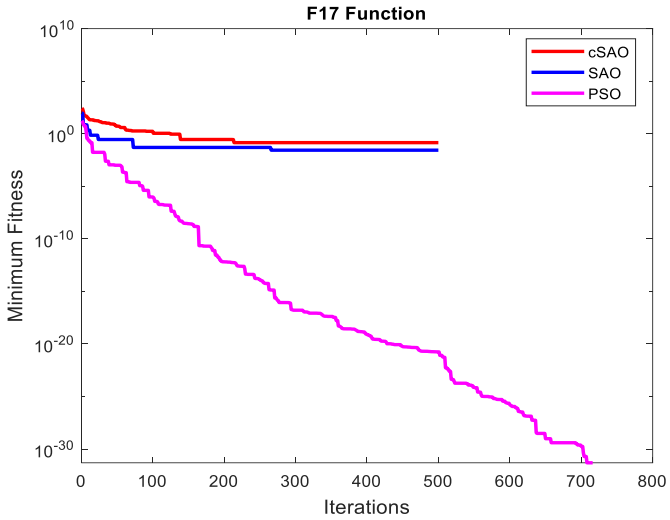

Fig. 20. Convergence curve for Rosenbrock function

5. Conclusion and Recommendation

5.1. Conclusion

This paper presented a new metaheuristic algorithm known as chaotic smell agent optimization (CSAO) that uses the concept of chaotic map in each of the modes of SAO. The CSAO operates in three distinct modes known as chaotic sniffing, trailing and random mode just like the original SAO. The chaotic maps introduced in each of the modes aid the search process and improve the performance of the algorithm. The performance of cSAO was evaluated on seventeen benchmark functions with different properties. Statistical results obtained showed that the cSAO outperformed the SAO and PSO with a final rank value of 1.33, 1.66 and 1.87 respectively. The convergence analysis also showed that cSAO converges faster than SAO by 25%. The chaotic SAO is easy to codify as the chaotic map equations are simple and easy to implement unlike other algorithms. The cSAO was aided by the chaotic maps which, help the algorithm perform high-speed searching and improve the tendency of being stuck in the local optima. The cSAO also has two major control parameters m and T which when carefully chosen make the algorithm suitable for different kinds of optimization problems.

5.2. Recommendation

cSAO can be applied to different areas of application such as speed regulation of a DC motor, image and signal processing, ball and plate system, control system design, transportation problem, task scheduling, data clustering, wireless and sensor network etc. Hybridization of the cSAO algorithm with other algorithms could also be considered to further enhance its performance. The convergence of the algorithm could also be considered as the algorithm has high exploration ability as a result of its number of modes when compared to other algorithms with only a single mode.

Acknowledgment

This work was financially supported by the Ahmadu Bello University, Zaria, Nigeria and University of Jos, Nigeria.

References

[1] S. Mirjalili, S. Mirjalili, and A. Lewis, "Grey Wolf Optimizer Adv Eng Softw 69: 46–61," ed: ed, 2014.

[2] O. Olorunda and A. P. Engelbrecht, "Measuring exploration/exploitation in particle swarms using swarm diversity," in 2008 IEEE congress on evolutionary computation (IEEE world congress on computational intelligence), 2008: IEEE, pp. 1128-1134.

[3] L. Lin and M. Gen, "Auto-tuning strategy for evolutionary algorithms: balancing between exploration and exploitation," Soft Computing, vol. 13, pp. 157-168, 2009.

[4] P. Agrawal, H. F. Abutarboush, T. Ganesh, and A. W. Mohamed, "Metaheuristic algorithms on feature selection: A survey of one decade of research (2009-2019)," IEEE Access, vol. 9, pp. 26766-26791, 2021.

[5] A. W. Mohamed, A. A. Hadi, and A. K. Mohamed, "Gaining-sharing knowledge based algorithm for solving optimization problems: a novel nature-inspired algorithm," International Journal of Machine Learning and Cybernetics, vol. 11, no. 7, pp. 1501-1529, 2020.

[6] G. Atali, İ. PehlIvan, B. Gürevin, and H. Seker, Ibrahim., "Chaos in metaheuristic based artificial intelligence algorithms: a short review," Turkish Journal of Electrical Engineering and Computer Sciences, vol. 29, no. 3, pp. 1354-1367, 2021.

[7] A. T. Salawudeen, M. B. Mu'azu, A. Yusuf, and A. E. Adedokun, "A Novel Smell Agent Optimization (SAO): An extensive CEC study and engineering application," Knowledge-Based Systems, vol. 232, p. 107486, 2021.

[8] A. T. Salawudeen et al., "Recent metaheuristics analysis of path planning optimaztion problems," in 2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS), 2020: IEEE, pp. 1-7.

[9] O. A. Meadows, M. B. Mu'Azu, and A. T. Salawudeen, "A Smell Agent Optimization Approach to Capacitated Vehicle Routing Problem for Solid Waste Collection," in 2022 IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development (NIGERCON), 2022: IEEE, pp. 1-5.

[10] S. Vishnoi, S. Nikolovski, M. Raju, M. K. Kirar, A. S. Rana, and P. Kumar, "Frequency Stabilization in an Interconnected Micro-Grid Using Smell Agent Optimization Algorithm-Tuned Classical Controllers Considering Electric Vehicles and Wind Turbines," Energies, vol. 16, no. 6, p. 2913, 2023.

[11] A. T. Salawudeen, A. A. Olaniyan, G. A. Olarinoye, and T. H. Sikiru, "Formulation and Optimization of Overcurrent Relay Coordination in Distribution Networks Using Metaheuristic Algorithms," in Information and Communication Technology and Applications: Third International Conference, ICTA 2020, Minna, Nigeria, November 24–27, 2020, Revised Selected Papers 3, 2021: Springer, pp. 389-402.

[12] S. Khan, S. Yang, and O. Ur Rehman, "A global particle swarm optimization algorithm applied to electromagnetic design problem," International Journal of Applied Electromagnetics and Mechanics, vol. 53, no. 3, pp. 451-467, 2017.

[13] S. Khan, M. Kamran, O. U. Rehman, L. Liu, and S. Yang, "A modified PSO algorithm with dynamic parameters for solving complex engineering design problem," International Journal of Computer Mathematics, vol. 95, no. 11, pp. 2308-2329, 2018/11/02 2018, doi: 10.1080/00207160.2017.1387252.

[14] A. Kumar, B. KUMAR SINGH, and B. Patro, "Diversity Preserving Auto Improved–PSO for Solving Optimization Problems," Journal of Multiple-Valued Logic & Soft Computing, vol. 29, no. 6, 2017.

[15] A. Salawudeen, M. Mu'azu, Y. Sha'aban, and E. Adedokun, "On the development of a novel smell agent optimization (SAO) for optimization problems," in 2nd International Conference on Information and Communication Technology and its Applications (ICTA 2018), Minna, 2018.

[16] M. Kohli and S. Arora, "Chaotic grey wolf optimization algorithm for constrained optimization problems," Journal of Computational Design and Engineering, vol. 5, no. 4, pp. 458-472, 2017, doi: 10.1016/j.jcde.2017.02.005.

[17] S. E. Jorgensen and B. Fath, Encyclopedia of ecology. Newnes, 2014.

[18] Manuel, Melanie. "The butterfly effect." PhD diss., faculty of the School of Education in partial fulfillment of the requirements for the degree Doctor of Education in the Department of Curriculum and Instruction, Indiana University, 2025.

[19] S. Arora, M. Sharma, and P. Anand, "A novel chaotic interior search algorithm for global optimization and feature selection," Applied Artificial Intelligence, vol. 34, no. 4, pp. 292-328, 2020.

[20] E. Varol Altay and B. Alatas, "Bird swarm algorithms with chaotic mapping," Artificial Intelligence Review, vol. 53, no. 2, pp. 1373-1414, 2020.