Journal of Computational Design and Engineering 5 (2018) 275–284

Contents lists available at ScienceDirect

Journal of Computational Design and Engineering

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

Chaotic whale optimization algorithm

Gaganpreet Kaur, Sankalap Arora *

DAV University, Jalandhar, Punjab, India

ARTICLE INFO

Article history:
Received 7 July 2017
Received in revised form 13 September 2017
Accepted 28 December 2017
Available online 3 January 2018

Keywords:
Whale Optimization Algorithm
Meta-heuristic algorithm
Chaos
Chaotic maps

ABSTRACT

The Whale Optimization Algorithm (WOA) is a recently developed meta-heuristic optimization algorithm which is based on the hunting mechanism of humpback whales. Similarly to other meta-heuristic algorithms, the main problem faced by WOA is slow convergence speed. So to enhance the global convergence speed and to get better performance, this paper introduces chaos theory into WOA optimization process. Various chaotic maps are considered in the proposed chaotic WOA (CWOA) methods for tuning the main parameter of WOA which helps in controlling exploration and exploitation. The proposed CWOA methods are benchmarked on twenty well-known test functions. The results prove that the chaotic maps (especially Tent map) are able to improve the performance of WOA.

1. Introduction

In many optimization problems, it is required to find the optimal solution to a given problem under highly complex constraints in a reasonable amount of time. Generally, modern intelligent methods are used to deal with these types of optimization problems. There are various methods that are proposed in order to solve these problems but they are insufficient to produce better results. In the past few decades, meta-heuristic optimization algorithms have achieved a lot of attention in scientific communities with significant developments, especially for solving many complex optimization problems. Prior to meta-heuristic algorithms, Hill-Climbing, Random Search and Simulated Annealing (SA), were the traditional algorithms used to solve the optimization problems (Yang, 2010a, 2010b, 2010c). Traditional algorithms start their search from a single point and require gradient information that consumed a lot of time to reach the global optima (Sivanandam & Deepa, 2007). Due to their limited pertinence and intricacy of constraints, these algorithms were not very effective for solving real world applications like localization problem (Arora & Singh, 2017b), economical optimization (Gao, Wang, Ovaska, & Xu, 2010), structural optimization problems (Gandomi, Yang, Talatahari, & Alavi, 2013) and engineering design problems (Coello, 2000) which involve different constraints to be gratified.

Basically, meta-heuristic algorithms impersonate biological or physical phenomenon to handle complex real world optimization problems. Unlike classical techniques, these meta-heuristic algorithms are mostly derivation-free (Yang, 2010a, 2010b, 2010c). Due to their stochastic nature, meta-heuristic algorithms have superior abilities to avoid local optima entrapment. These algorithms can be applied to various fields due to their simplicity, flexibility, robustness, and efficiency (Sivanandam & Deepa, 2007). Some of the most prominent nature inspired meta-heuristic algorithms developed so far are Particle Swarm Optimization (PSO) (Eberhart & Kennedy, 1995; Kennedy, 2011), Artificial Bee Colony (ABC) (Dorigo & Di Caro, 1999; Dorigo & Gambardella, 1997), Firefly Algorithm (FA) (Yang, 2010a, 2010b), Biogeography-Based Optimization algorithm (BBO) (Simon, 2008), Bat Algorithm (Gandomi et al., 2013; Tsai et al., 2015), Krill-Herd (KH) (Gandomi & Alavi, 2012), Cat Swarm Optimization (Shu-Chuan, Pei-Wei, & Jeng-Shyang, 2006), Grey Wolf Optimizer (GWO) (Mirjalili, Mirjalili, & Lewis, 2014), Ant Lion Optimizer (ALO) (Mirjalili, 2015), Butterfly Optimization Algorithm (BOA) (Arora & Singh, 2015) and most recently Whale Optimization Algorithm (WOA) (Mirjalili & Lewis, 2016).

Above all, the most challenging task encountered in the development of any meta-heuristic algorithm is to find a proper balance between exploration and exploitation due to the stochastic nature of the optimization process (Mirjalili & Lewis, 2016). The exploration phase helps the optimizer to globally explore the search space as extensively as possible. Also, the population faces some abrupt changes in this phase. In contrast, the exploitation phase involves the refinement of the promising solutions obtained from the exploration phase. Here, the population encounters small abrupt changes (Alba & Dorronsoro, 2005).

WOA is a recently developed nature-inspired meta-heuristic that imitates the social behavior of humpback whales (Mirjalili & Lewis,

Peer review under responsibility of Society for Computational Design and Engineering.

* Corresponding author.
E-mail addresses: gaganpreet1292@gmail.com (G. Kaur), sankalap.arora@gmail.com (S. Arora).

https://doi.org/10.1016/j.jcde.2017.12.006
2288-4300/

2016). This algorithm is inspired by the bubble-net foraging method of the humpback whales. The method includes three main steps of hunting, i.e., encircling prey, searching for prey and attacking the prey. It has been proved that this algorithm is able to show very competitive results compared to other meta-heuristic algorithms in solving various real world problems like optimal sizing of renewable resources for loss reduction in distribution systems (Reddy, Reddy, & Manohar, 2017), feature selection (Mafarja & Mirjalili, 2017), sizing optimization for skeletal structures (Kaveh, 2017) and data clustering (Jadhav & Gomathi, 2017). Identical to other meta-heuristic algorithms slow convergence speed is the main problem encountered by WOA (Aljarah et al. 2016). To further enhance the performance this paper introduces chaos theory into WOA.

With the development of the non-linear dynamics chaos theory has been extensively utilized in various applications (Pecora & Carroll, 1990). Chaos theory is related to the study of chaotic dynamical systems that are highly sensitive to initial conditions and includes infinite unstable periodic motions. In order to improve performance, chaos has been employed in various meta-heuristic algorithms, which results in better convergence speed and avoidance from local optima entrapment (Kellert, 1994). Although it appears to be stochastic, providing chaotic behavior does not crucially need stochasticity. Deterministic systems are also able to show chaotic behaviors. Earlier, chaos theory has been utilized by various meta-heuristic algorithms such as genetic algorithm (Li-Jiang & Tian-Lun, 2002), harmony search (Alatas, 2010a, 2010b), PSO (Liu, Wang, Jin, Tang, & Huang, 2005), ABC (Alatas, 2010a, 2010b), FA, KH (Wang, Guo, Gandomi, Hao, & Wang, 2014), BOA (Arora & Singh, 2017a) and GWO (Kohli & Arora, 2017) to enhance the performance of the algorithms by tuning certain parameters.

The aim of this paper is to introduce Chaotic Whale Optimization Algorithm (CWOA) based methods in which different chaotic systems are used to replace the critical parameter of WOA that helps to switch the local and global searching ability of WOA. A subset of unimodal and multimodal benchmark functions have been employed in order to evaluate the proposed CWOA.

The rest of the paper is organized as follows. Review of WOA is presented in Section 2. The chaotic maps that describe chaotic sequences for WOA are described in Section 3. In Section 4, the proposed CWOA have been presented. The experimental results have been described in Section 5. Finally, the conclusions and future work have been discussed in Section 6.

2. Whale optimization algorithm

WOA was introduced by Mirjalili in the year 2016. This algorithm is a simulation of the hunting mechanism of humpback whales. This special hunting mechanism is called as bubble-net foraging method. This foraging is done by creating distinctive bubbles along a circular or '9-shaped' path while the humpback whales are encircling the prey. With the help of special maneuvers, the humpback whales can dive around 10–15 m down and then start creating bubbles in a spiral shape around the prey and swim up towards the surface. The humpback whales encircle the prey with flashing fins, which keeps the prey contained and prevent it from escaping (Mirjalili & Lewis, 2016). The mathematical model of encircling prey, spiral bubble-net foraging maneuver and search for prey is described in the following section:

2.1. Encircling prey

Humpback whales encircle the prey and update their position towards the best search agent with the increasing number of iterations from start to a maximum number of iterations. This behavior is mathematically formulated as:

$$\vec{D} = |\vec{C} \cdot \overrightarrow{X^*}(t) - \vec{X}(t)| \tag{1}$$

$$\vec{X}(t+1) = \overrightarrow{X^*}(t) - \vec{A} \cdot \vec{D} \tag{2}$$

where $\vec{A}$ and $\vec{C}$ are the coefficient vectors, t indicates the current iteration, X^* is the position vector of the best solution obtained so far, $\vec{X}$ is the position vector, || is the absolute value and . is an element-by-element multiplication. The vectors $\vec{A}$ and $\vec{C}$ are calculated as follows:

$$\vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a} \tag{3}$$

$$\vec{C} = 2 \cdot \vec{r} \tag{4}$$

where $\vec{a}$ is linearly decreased from 2 to 0 over the course of iterations (in both exploration and exploitation phases) and $\vec{r}$ is a random vector in [0, 1].

2.2. Bubble-net attacking method

The following two approaches are designed in order to mathematically model the bubble-net behavior of the humpback whales:

1. *Shrinking encircling mechanism:* This behavior is achieved by decreasing the value of a from 2 to 0 in Eq. (3) over the course of iterations. The new position of a search agent can be defined anywhere in between the original position of the agent and the position of the current best agent by setting random values for $\vec{A}$ in $[-1, 1]$.

2. *Spiral updating position:* The spiral equation between the position of the prey and the whale to imitate the helix-shaped movement of the humpback whales is as follows:

$$\vec{X}(t+1) = \vec{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X^*}(t) \tag{5}$$

The pseudocode of the WOA algorithm is illustrated in Algorithm 1.

Algorithm 1. Pseudo-code of the WOA algorithm

```
Initialize the whales population Xi (i = 1, 2,..., n)
Calculate the fitness of each search agent
X* = the best search agent
while (t < maximum number of iterations)
  for each search agent
    Update a, A, C, l and p
      if1 (p < 0.5)
        if2(|A| < 1)
          Update the position of the current search agent by
   the Eq. (1)
        else if2 (|A| ≥ 1)
          Select a random search agent (Xrand)
            Update the position of the current search agent
   by Eq. (8)
        end if2
          else if1 (p ≥ 0.5)
            Update the position of the current search by the
   Eq. (5)
      end if1
  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*
```

Table 1
Chaotic maps.

S. no.	Name	Chaotic map
1.	Logistic	$x_{i+1} = ax_i(1 - x_i)$
2.	Cubic	$x_{i+1} = ax_i(1 - x_i^2)$
3.	Sine	$x_{i+1} = \frac{a}{4}\sin(\pi x_i)$
4.	Sinusoidal	$x_{i+1} = ax_i^2\sin(\pi x_i)$
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.	Circle	$x_{i+1} = \mathrm{mod}(x_i + b - (\frac{a}{2\pi})\sin(2\pi x_k), 1)$
7.	Iterative	$x_{i+1} = \sin(a\pi/x_i)$
8.	Tent	$x_{i+1} = \begin{cases} \frac{x_i}{0.7} & x_i < 0.7 \\ \frac{10}{3}(1 - x_i) & x_i \geqslant 0.7 \end{cases}$
9.	Piecewise	$x_{i+1} = \begin{cases} \frac{x_i}{P} & 0 \leq x_i < P \\ \frac{x_i - P}{0.5 - P} & P \leq x_i < 0.5 \\ \frac{1 - P - x_i}{0.5 - P} & 0.5 \leq x_i < 1 - P \\ \frac{1 - x_i}{P} & 1 - P \leq x_i < 1 \end{cases}$
10.	Gauss/mouse	$x_{i+1} = \begin{cases} 1 & x_i = 0 \\ \frac{1}{\mathrm{mod}(x_i, 1)} & \text{otherwise} \end{cases}$

It is worth noticing that humpback whales swim around the prey within a shrinking circle and along a spiral-shaped path simultaneously. So in order to model this behavior, we assume the probability of 50% to choose between either the shrinking encircling method or the spiral model, to update the position of whales. The mathematical model is described as follows:

$$\vec{X}(t+1) = \begin{cases} \overrightarrow{X^*}(t) - \vec{A}.\vec{D}\} & \text{if} \quad p \leqslant 0.5 \\ \vec{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X^*}(t)\} & \text{if} \quad p \geqslant 0.5 \end{cases} \tag{6}$$

where $\vec{D'} = |\overrightarrow{X^*}(t) - \vec{X}(t)|$ and indicates the distance of the ith whale to the prey (the best solution obtained so far), b is constant for defining the shape of the logarithmic spiral, l is a random number in [−1, 1] and p represents a random number in [0, 1].

2.3. Search for prey

The variation of $\vec{A}$ vector can be utilized to search for prey, i.e., exploration phase. Therefore, $\vec{A}$ can be used with the random values greater than 1 or less than −1 to force search agents to move away from a reference whale. The mathematical model for this phase is as follows:

$$\vec{D} = |\vec{C} \cdot \overrightarrow{X_{rand}} - \vec{X}| \tag{7}$$

$$\vec{X}(t+1) = \overrightarrow{X_{rand}} - \vec{A} \cdot \vec{D} \tag{8}$$

where $\overrightarrow{X_{rand}}$ Is a random position vector (a random whale) chosen from the current population.

3. Chaotic maps

In this section, we present ten 1-D non-invertible chaotic maps that are utilized to produce chaotic sets. This set of chaotic maps with initial point 0.7 has been chosen with different behaviors. The initial point can be chosen any number between 0 and 1 (or depends on the range of chaotic map). However, it should be noted that the initial value may have significant impacts on the fluctuation pattern of some of the chaotic maps (Saremi, Mirjalili, & Lewis, 2014). Recently, a lot of chaotic maps were discovered primarily by mathematicians and physicians applicable to different domains of human activity. In line with this, the majority of these were applied to different algorithms for solving the various real-world problems. Therefore, the more applicable chaotic maps tackling the optimization algorithms are used in the current research work as shown in Table 1 (Gandomi & Yang, 2014; Gandomi, Yang, Alavi, & Talatahari, 2013; Kohli & Arora, 2017).

4. Chaotic whale optimization algorithm (CWOA)

In spite of having good convergence rate, WOA still cannot perform that better in finding the global optima which affect the convergence rate of the algorithm. So, to reduce this affect and to improve its efficiency, CWOA algorithm is developed by introducing chaos in WOA algorithm itself. Generally, chaotic comes from the word 'chaos' which means the property of a complex system whose behavior is so unpredictable, and map means mapping or associating chaos behavior in the algorithm with some parameter using a function. Due to the ergodicity and non-repetition properties of chaos, it can perform overall searches at higher speeds compared to the stochastic searches that basically rely on probabilities (Dos Santos Coelho & Mariani, 2008). Chaotic maps are the maps which show complex and dynamic behavior in the non-linear systems (Pecora & Carroll, 1990).

Due to their dynamic behavior, chaotic maps have been widely acknowledged in the field of optimization which helps optimization algorithm in exploring the search space more vigorously and globally (Yang, Li, & Cheng, 2007). In almost all meta-heuristic algorithms with stochastic components, randomness is achieved by using probability distributions. It can be advantageous to replace such randomness by using chaotic maps. In order to introduce chaos in optimization algorithm, different chaotic maps having different mathematical equations are used that are listed in Table 1. Since last decade, chaotic maps have been extensively appreciated in the field of optimization algorithms in

Algorithm 2. Pseudo-code of CWOA algorithm

Initialize the generation counter t and randomly initialize the whale's population X_i (i = 1, 2,. . ., n)
Evaluate the fitness of each search agent to find the best search agent X^*
Initialize the value of the chaotic map x_0 randomly
while (t < maximum number of iterations)
  Update the chaotic number using the respective chaotic map equation
    for each search agent
      Update *a, A, C, l* and *p*
        if1 (p < 0.5)
          if2(|A| < 1)
            Update the position of the current search agent by the Eq. (1)
          else if2 (|A| ≥ 1)
            Select a random search agent (*Xrand*)
            Update the position of the current search agent by Eq. (8)
        end if2
          else if1 ($p \geq 0.5$)
            Update the position of the current search by the Eq. (5)
      end if1
  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^*

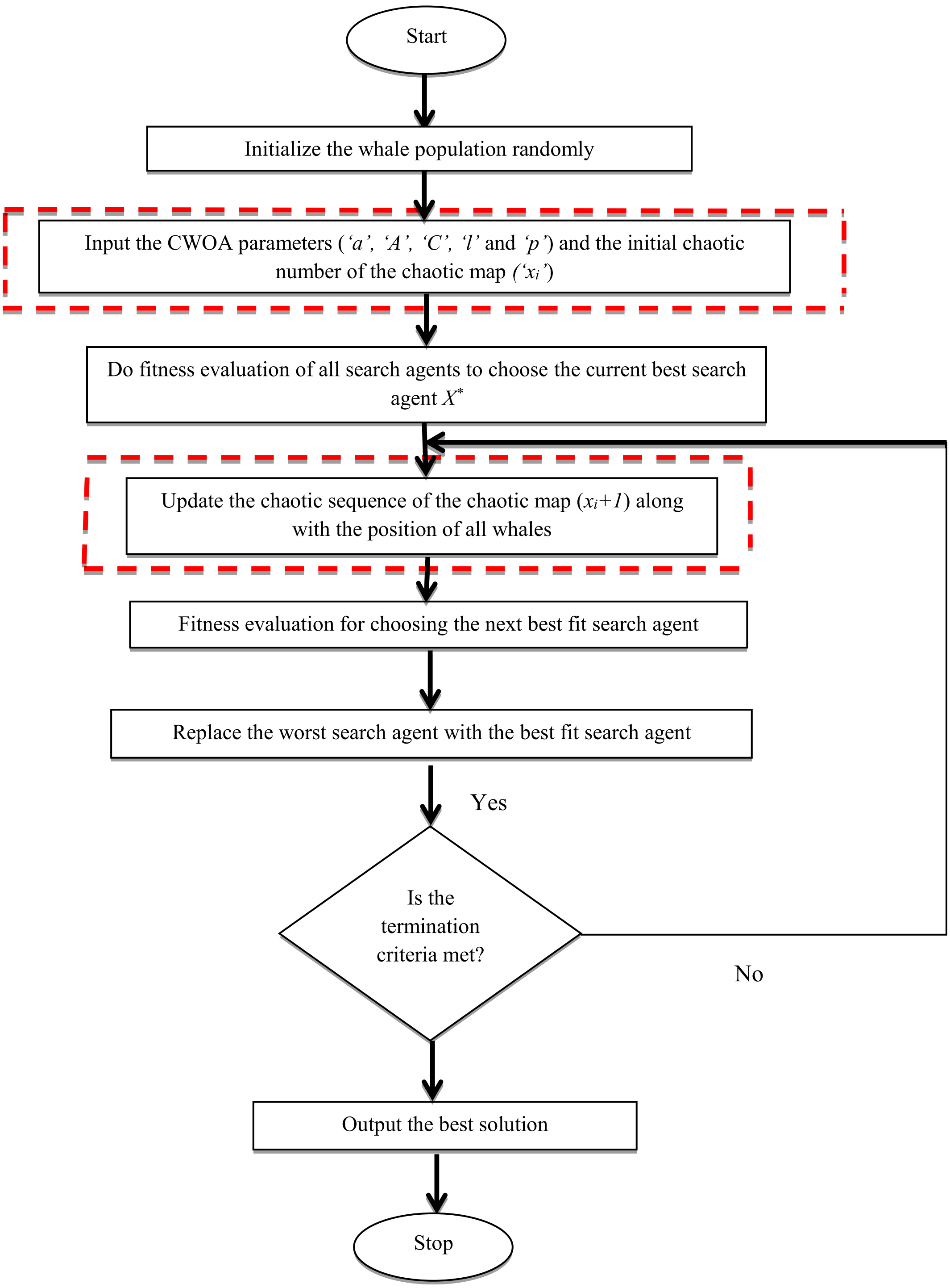

Fig. 1. Flowchart of optimization procedure of CWOA.

exploring the search space more dynamically and globally. In consonance with different human's domain a large variation of chaotic maps designed by physicians, researchers and mathematicians are available in the optimization field (He, He, Jiang, Zhu, & Hu, 2001). Out of all these, ten most significant uni-dimensional chaotic maps (Gandomi & Yang, 2014) have been employed in the present work to tackle CWOA, details of which are given in Table 1.

The convergence rate of WOA has been positively influenced by utilizing chaotic maps as these maps persuade chaos in the feasible region which is predicted only for very short initial time and is stochastic for a longer period of time (Yang, Li, & Cheng, 2007). Pseudocode of the proposed CWOA algorithm is illustrated in Algorithm 2.

The optimization procedure of the proposed CWOA is also presented in the form of flowchart given in Fig. 1. The very first step of the flowchart implicates the stochastic initialization of population of whales. Then, a respective chaotic map is chosen to be mapped with the algorithm along with the initialization of its first chaotic number and a variable (Gandomi & Yang, 2014). After this the parameters of the CWOA algorithm involved in controlling the

Table 2
Benchmark functions used in the study.

No.	Benchmark function	Formula	Dim	Range	Optimal 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	$f_x = \sum_{i=0}^{n-1} (\lvert x_i \rvert + 0.5)^2$	30	[−100, 100]	0
F5	Quartic Noise	$f_x = \sum_{i=0}^{n-1} x_i^4 + N(0,1)$	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	$f_x = -20\exp(0.02)\sqrt{1}/D\sum_{i=1}^{D} x_i^2 - \exp(1/D\sum_{i=1}^{d}\cos(2\pi(x_i) + 20 + \exp$	30	[−32.768, 32.768]	0
F8	Griewank	$f_x = \frac{1}{4000}\sum_{i=0}^{n-1}(x_i - 100)^2 - \prod_{i=1}^{n-1}\cos(\frac{(x_i-100)}{\sqrt{i-1}}) + 1$	30	[−600, 600]	0
F9	Levy	$f_x = \sin2(\pi\omega1) + \sum_{i=1}^{d-1}(\omega i - 1)2[1 + 10\sin2(\pi\omega i + 1)] + (\omega d - 1)2[1 + \sin2(2\pi\omega d)]$	30	[−10, 10]	0
F10	Michalewitz	$f_x = -\sum_{i=0}^{n}(\sin(x_i)\sin^{20}(\frac{ix_i^2}{\pi})^{2m}$	10	[−100, 100]	0.966n
F11	Rastrigin	$f_x = (x_i^2 - 10\cos 2\pi x_i + 10)$	30	[−5.12, 5.12]	0
F12	Alpine	$f_x = \sum_{i=0}^{n}\lvert x_i \sin x_i + 0.1x_i \rvert$	30	[−100, 100]	0
F13	Schaffer	$f_x = (x_i^2 + x_{i+1}^2)^{\frac{1}{4}}(50(x_i^2 + x_{i+1}^2)^{0.1} + 1)$	2	[−100, 100]	0
F14	Rosenbrock	$f_x = 100(x_{i+1} - x_i^2)^2 + (1.0 - x_i)^2$	30	[−10, 10]	0
F15	Easom	$f_x - \cos(x_i)\cos(x_{i+1})\exp(-(x_i - \Omega)^2 - (x_{i+1} - \Omega)^2)$	2	[−100, 100]	−1
F16	Shubert	$f_x = (\sum_{i=1}^{5} i\cos(i+1)x_i + i)(\sum_{i=1}^{5} i\cos(i+1)x_{i+1} + i)$	2	[−10, 10]	−186.73
F17	Schwefel 1.2	$f_x = \sum_{i=0}^{n-1}\{\sum_{j=0}^{j<1} x_i\}2$	30	[−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	[−500, 500]	−12569.5

exploration and exploitation mechanism specifically *a, A, C, l* and *p* are initialized which are same as in WOA. Also, the chaotic number of the chaotic map is initialized to adjust the parameter '*p*' of WOA which is highlighted in Fig. 1. In the next step, fitness of all the whales initialized in the search space is evaluated using the various standard benchmark functions. The whale with the highest fitness is assumed to be the current best search agent. The current best search agent will keep updating its position using Eq. (1), when the value of control parameter $A < 1$. Similarly, when the value of parameter $A \geq 1$, a random whale is chosen and the position of the current best search agent is updated using Eq. (8) if there is a new best search agent than the last one. Sequentially the fitter whale will keep updating its position and at the end may get the first position as optimal solution. The value of parameter '*p*' is also updated along with the course of iterations using Eqs. (3) and (4). At the end of the last iteration, the best search agent will be considered as the most optimal solution by the CWOA algorithm.

5. Experimental results and discussions

In order to measure and test the performance of every novel optimization algorithm, the algorithm must deal with some well-defined test functions. In this section, various experiments on optimization benchmark problems are implemented to verify the performance of the proposed meta-heuristic CWOA method. Twenty well-known benchmark functions (Digalakis & Margaritis, 2001; Yao, Liu, & Lin, 1999) have been utilized in order to check the performance of CWOA. These functions are divided into two categories: unimodal and multimodal. Unimodal benchmark functions have single optima and they are well suited for benchmarking exploitation. In contrast, multimodal benchmark functions have more than one optima that makes them more challenging than unimodal functions. One of the optima is called global optima and the rest are called local optima. Avoiding the local optima and determining the global optimum should be the main characteristics of any powerful meta-heuristic algorithm. Therefore, the multimodal benchmark functions are responsible for testing exploration and avoiding the entrapment in local optima. Note that the minima of most of the unimodal and multimodal benchmark functions is 0 except some functions, i.e., F10, F15, F16 and F20. The properties of unimodal and multimodal benchmark functions are listed in Table 2, where *Dim* indicates the dimension of the function, and *Range* is the boundary of the function's search space. The performance of CWOA with different chaotic maps and their results have been discussed in Section 5.1. Also, the qualitative analysis and the statistical testing of the results have been clearly described in Sections 5.2 and 5.3 respectively.

5.1. The performance of CWOA with different chaotic maps

For the results of various CWOA's, the population size of the whales is taken 30 and 50 iterations are performed. The results are averaged over thirty independent runs. CWOA1 to CWOA10 utilize Logistic, Cubic, Sine, Sinusoidal, Singer, Circle, Iterative, Tent, Piecewise and Gauss/mouse maps, respectively as shown in Table 1. It can be seen from Table 3 that CWOA6 and CWOA7 algorithms show worse results as compared to WOA algorithm. This shows that, Circle and Iterative chaotic maps are not able to improve the performance of WOA algorithm. In contrast, CWOA1, CWOA2, CWOA3, CWOA4, CWOA5, CWOA8, CWOA9 and CWOA10 algorithms show much better results as compared to WOA algorithm. In other words, Logistic, Cubic, Sine, Sinusoidal, Singer, Tent, Piecewise and Gauss/mouse chaotic maps are able to enhance the performance of WOA algorithm successfully. From the results of Table 3, it is proved that the Tent–based WOA algorithm yields the best results on all the test functions. The *p* values depicted that this supremacy is statistically significant. It can also be seen from the table that the Cubic, Sine, Piecewise and Tent chaotic maps yield the best results in more than thirteen test functions. On the other hand, Circle and Iterative maps have provided worse results in more than thirteen test functions.

5.2. Qualitative analysis

Qualitative analysis on different benchmark functions for further effective evaluation of the performance of CWOA has also

Table 3
Results of 10 chaotic maps on all benchmark functions on CWOA.

Sphere	Mean	Std. Dev.	*p* Values	Beale	Mean	Std. Dev.	*p* Values	Cigar	Mean	Std. Dev.	*p* Values
WOA	2.58E−52	7.98E−52	0.005	WOA	1.62E−01	3.41E−01	0.386	WOA	7.82E−57	1.83E−56	0.005
CWOA1	3.00E−69	6.10E−69	0.445	CWOA1	4.14E−03	6.46E−03	0.241	CWOA1	2.49E−71	7.77E−71	0.114
CWOA2	1.11E−67	3.51E−67	0.799	CWOA2	9.93E−03	2.49E−02	0.093	CWOA2	8.57E−71	2.52E−70	0.059
CWOA3	1.80E−70	3.41E−70	0.959	CWOA3	**8.94E−04**	**8.36E−04**	**N/A**	CWOA3	4.63E−72	1.28E−71	0.285
CWOA4	2.00E−69	5.606E−69	0.721	CWOA4	5.57E−03	1.44E−02	0.878	CWOA4	1.68E−72	4.97E−72	0.386
CWOA5	2.95E−70	4.45E−70	0.721	CWOA5	5.83E−01	5.16E−01	0.005	CWOA5	5.49E−74	1.58E−73	0.575
CWOA6	1.64E+08	2.23E+08	0.005	CWOA6	7.24E−01	2.32E−01	0.005	CWOA6	1.26E+04	8.88E+03	0.005
CWOA7	1.14E+08	1.43E+08	0.005	CWOA7	7.24E−01	2.32E−01	0.005	CWOA7	1.07E+04	5.56E+03	0.005
CWOA8	3.26E−70	7.52E−70	0.445	CWOA8	2.24E−01	4.67E−01	0.028	CWOA8	**4.64E−74**	**1.26E−73**	**N/A**
CWOA9	**1.68E−70**	**2.58E−70**	**N/A**	CWOA9	1.80E−01	3.68E−01	0.013	CWOA9	2.59E−73	7.42E−73	0.445
CWOA10	2.78E−70	7.71E−70	0.508	CWOA10	9.75E−01	3.98E−01	0.005	CWOA10	5.26E−73	1.23E−72	0.575
Step	Mean	Std. Dev.	*p* Values	Quartic Noise	Mean	Std. Dev.	*p* Values	Bohachevsky	Mean	Std. Dev.	*p* Values
WOA	**0.00E+00**	**0.00E+00**	**N/A**	WOA	3.05E−02	3.37E−02	0.799	WOA	**−5.55E−17**	**0.00E+00**	**N/A**
CWOA1	**0.00E+00**	**0.00E+00**	**N/A**	CWOA1	4.08E−02	4.59E−02	0.386	CWOA1	**−5.55E−17**	**0.00E+00**	**N/A**
CWOA2	**0.00E+00**	**0.00E+00**	**N/A**	CWOA2	3.60E−02	2.64E−02	0.059	CWOA2	**−5.55E−17**	**0.00E+00**	**N/A**
CWOA3	**0.00E+00**	**0.00E+00**	**N/A**	CWOA3	3.34E−02	2.39E−02	0.169	CWOA3	**−5.55E−17**	**0.00E+00**	**N/A**
CWOA4	**0.00E+00**	**0.00E+00**	**N/A**	CWOA4	1.95E−02	1.83E−02	0.878	CWOA4	**−5.55E−17**	**0.00E+00**	**N/A**
CWOA5	**0.00E+00**	**0.00E+00**	**N/A**	CWOA5	3.09E−02	2.50E−02	0.386	CWOA5	**−5.55E−17**	**0.00E+00**	**N/A**
CWOA6	2.91E+04	1.40E+04	0.005	CWOA6	2.40E+00	1.12E+00	0.005	CWOA6	1.76E−01	3.09E−01	0.109
CWOA7	2.81E+04	6.24E+03	0.005	CWOA7	1.82E+00	6.20E−01	0.005	CWOA7	1.76E−01	3.09E−01	0.043
CWOA8	**0.00E+00**	**0.00E+00**	**N/A**	CWOA8	**1.91E−02**	**1.54E−02**	**N/A**	CWOA8	**−5.55E−17**	**0.00E+00**	**N/A**
CWOA9	**0.00E+00**	**0.00E+00**	**N/A**	CWOA9	4.47E−02	3.51E−02	0.047	CWOA9	**−5.55E−17**	**6.13E−21**	**N/A**
CWOA10	**0.00E+00**	**0.00E+00**	**N/A**	CWOA10	2.11E−02	2.99E−02	0.575	CWOA10	**−5.55E−17**	**0.00E+00**	**N/A**
Ackley	Mean	Std. Dev.	*p* Values	Greiwank	Mean	Std. Dev.	*p* Values	Levy	Mean	Std. Dev.	*p* Values
WOA	**−1.44E−16**	**0.00E+00**	**N/A**	WOA	**0.00E+00**	**0.00E+00**	**N/A**	WOA	**2.24E+00**	**4.58E−01**	**N/A**
CWOA1	**−1.44E−16**	**0.00E+00**	**N/A**	CWOA1	**0.00E+00**	**0.00E+00**	**N/A**	CWOA1	2.37E+00	9.38E−01	0.003
CWOA2	**−1.44E−16**	**0.00E+00**	**N/A**	CWOA2	**0.00E+00**	**0.00E+00**	**N/A**	CWOA2	2.41E+00	3.73E−01	0.003
CWOA3	**−1.44E−16**	**9.86E−21**	**N/A**	CWOA3	**0.00E+00**	**0.00E+00**	**N/A**	CWOA3	2.37E+00	9.38E−01	0.003
CWOA4	−1.4E−16	**9.86E−21**	**N/A**	CWOA4	**0.00E+00**	**0.00E+00**	**N/A**	CWOA4	2.54E+00	4.94E−01	0.003
CWOA5	−1.4E−16	**8.05E−21**	**N/A**	CWOA5	**0.00E+00**	**0.00E+00**	**N/A**	CWOA5	2.63E+00	5.68E−01	0.003
CWOA6	1.81E+01	1.11E+00	0.005	CWOA6	1.41E+02	6.63E+01	0.005	CWOA6	2.41E+00	3.73E−01	0.003
CWOA7	1.78E+01	1.86E+00	0.005	CWOA7	1.04E+02	4.36E+01	0.005	CWOA7	6.25E+01	3.15E+01	0.003
CWOA8	**−1.44E−16**	**0.00E+00**	**N/A**	CWOA8	**0.00E+00**	**0.00E+00**	**N/A**	CWOA8	2.60E+00	3.55E−01	0.003
CWOA9	**−1.44E−16**	**0.00E+00**	**N/A**	CWOA9	**0.00E+00**	**0.00E+00**	**N/A**	CWOA9	2.45E+00	3.61E−01	0.003
CWOA10	**−1.44E−16**	**0.00E+00**	**N/A**	CWOA10	**0.00E+00**	**0.00E+00**	**N/A**	CWOA10	2.43E+00	4.18E−01	0.003
Michalewitz	Mean	Std. Dev.	*p* Values	Rastrigin	Mean	Std. Dev.	*p* Values	Alpine	Mean	Std. Dev.	*p* Values
WOA	−3.04E+00	7.21E−01	0.013	WOA	**0.00E+00**	**0.00E+00**	**N/A**	WOA	5.11E−31	8.75E−31	0.005
CWOA1	−2.98E+00	3.67E−01	0.005	CWOA1	**0.00E+00**	**0.00E+00**	**N/A**	CWOA1	9.74E−39	2.99E−38	0.386
CWOA2	−2.98E+00	7.03E−01	0.009	CWOA2	**0.00E+00**	**0.00E+00**	**N/A**	CWOA2	3.30E−39	9.78E−39	0.059
CWOA3	−2.96E+00	6.15E−01	0.005	CWOA3	**0.00E+00**	**0.00E+00**	**N/A**	CWOA3	3.30E−39	9.78E−39	0.646
CWOA4	−2.77E+00	5.21E−01	0.005	CWOA4	**0.00E+00**	**0.00E+00**	**N/A**	CWOA4	6.17E−40	1.44E−39	0.386
CWOA5	−3.02E+00	8.82E−01	0.009	CWOA5	**0.00E+00**	**0.00E+00**	**N/A**	CWOA5	**8.91E−42**	**1.06E−40**	**N/A**
CWOA6	−4.27E+00	7.78E−01	0.386	CWOA6	2.54E+02	4.56E+01	0.005	CWOA6	7.22E+03	2.15E+03	0.005
CWOA7	**−4.40E+00**	**8.75E−01**	**N/A**	CWOA7	2.55E+02	4.71E+01	0.005	CWOA7	6.76E+03	1.28E+03	0.005
CWOA8	−2.51E+00	5.05E−01	0.005	CWOA8	**0.00E+00**	**0.00E+00**	**N/A**	CWOA8	4.34E−39	1.33E−38	0.959
CWOA9	−2.94E+00	6.06E−01	0.007	CWOA9	**0.00E+00**	**0.00E+00**	**N/A**	CWOA9	3.95E−40	6.09E−40	0.203
CWOA10	−2.68E+00	7.53E−01	0.009	CWOA10	**0.00E+00**	**0.00E+00**	**N/A**	CWOA10	1.08E−39	2.58E−39	0.333
Schaffer	Mean	Std. Dev.	*p* Values	Rosenbrock	Mean	Std. Dev.	*p* Values	Easom	Mean	Std. Dev.	*p* Values
WOA	5.13E−26	9.66E−26	0.005	WOA	**2.88E+01**	**2.44E−02**	**N/A**	WOA	**−8.76E−01**	**3.09E−01**	**N/A**
CWOA1	4.37E−29	1.10E−28	0.005	CWOA1	**2.88E+01**	**4.59E−02**	**N/A**	CWOA1	−4.71E−01	4.99E−01	0.047
CWOA2	1.43E−28	3.62E−28	0.005	CWOA2	**2.88E+01**	**3.61E−02**	**N/A**	CWOA2	−3.33E−01	4.49E−01	0.028
CWOA3	5.17E−29	1.06E−28	0.005	CWOA3	**2.88E+01**	**2.48E−02**	**N/A**	CWOA3	−6.12E−01	4.49E−01	0.139
CWOA4	1.39E−28	4.39E−28	0.005	CWOA4	**2.88E+01**	**1.73E−02**	**N/A**	CWOA4	−6.07E−01	4.32E−01	0.047
CWOA5	4.05E−29	1.27E−28	0.005	CWOA5	**2.88E+01**	**2.91E−02**	**N/A**	CWOA5	−4.58E−01	4.92E−01	0.037
CWOA6	**4.04E−37**	**1.16E−36**	**N/A**	CWOA6	4.61E+05	5.17E+05	0.005	CWOA6	0.00E+00	0.00E+00	0.005
CWOA7	2.24E−36	7.05E−36	0.093	CWOA7	1.95E+05	2.18E+05	0.005	CWOA7	−2.00E−01	4.22E−01	0.028
CWOA8	2.19E−30	5.37E−30	0.005	CWOA8	**2.88E+01**	3.78E−02	0.317	CWOA8	−3.73E−01	4.84E−01	0.009
CWOA9	2.79E−31	4.49E−31	0.005	CWOA9	**2.88E+01**	1.45E−02	0.317	CWOA9	−3.70E−01	4.82E−01	0.059
CWOA10	8.37E−29	1.71E−28	0.005	CWOA10	**2.88E+01**	2.46E−02	0.317	CWOA10	−8.37E−01	3.01E−01	0.241
Shubert	Mean	Std. Dev.	*p* Values	Schwefel 1.2	Mean	Std. Dev.	*p* Values	Schwefel 2.21	Mean	Std. Dev.	*p* Values
WOA	−7.17E+01	1.42E+01	0.108	WOA	7.90E−184	0.00E+00	0.005	WOA	1.10E−18	3.48E−18	0.005
CWOA1	−6.09E+01	1.27E+01	0.059	CWOA1	9.85E−210	0.00E+00	0.005	CWOA1	2.97E−29	8.70E−29	0.959
CWOA2	−7.62E+01	1.12E+01	0.074	CWOA2	2.44E−206	0.00E+00	0.005	CWOA2	1.11E−31	2.31E−31	0.575

Table 3 (*continued*)

Sphere	Mean	Std. Dev.	*p* Values	Beale	Mean	Std. Dev.	*p* Values	Cigar	Mean	Std. Dev.	*p* Values
CWOA3	−7.18E+01	1.23E+01	0.074	CWOA3	2.76E−206	0.00E+00	0.005	CWOA3	4.07E−30	7.94E−30	0.017
CWOA4	−7.59E+01	1.42E+01	0.074	CWOA4	9.80E−210	0.00E+00	0.005	CWOA4	1.52E−30	3.93E−30	0.575
CWOA5	−7.24E+01	1.12E+01	0.074	CWOA5	4.51E−206	0.00E+00	0.005	CWOA5	1.42E−28	4.49E−28	0.169
CWOA6	**−7.84E+01**	**2.04E+01**	**N/A**	CWOA6	8.60E−290	0.00E+00	0.646	CWOA6	8.31E+01	1.06E+01	0.005
CWOA7	−7.64E+01	1.92E+01	1.000	CWOA7	**1.14E−292**	**0.00E+00**	**N/A**	CWOA7	8.63E+01	7.44E+00	0.005
CWOA8	−7.81E+01	1.01E+01	0.110	CWOA8	6.25E−208	0.00E+00	0.005	CWOA8	**1.07E−31**	**1.46E−31**	**N/A**
CWOA9	−6.92E+01	1.22E+01	0.074	CWOA9	2.58E−209	0.00E+00	0.005	CWOA9	1.11E−30	2.26E−30	0.333
CWOA10	−7.50E+01	9.88E+00	0.074	CWOA10	5.54E−206	0.00E+00	0.005	CWOA10	1.24E−30	2.52E−30	0.445

Schwefel 2.22	Mean	Std. Dev.	*p* Values	Schwefel 2.26	Mean	Std. Dev.	*p* Values
WOA	1.93E−32	4.96E−32	0.005	WOA	**−1.14E+04**	**1.62E+03**	**N/A**
CWOA1	1.07E−40	2.21E−40	0.241	CWOA1	−1.03E+04	−1.03E+04	0.314
CWOA2	**3.31E−41**	**7.46E−41**	**N/A**	CWOA2	−9.38E+03	2.72E+03	0.074
CWOA3	1.55E−39	4.31E−39	0.059	CWOA3	−9.35E+03	2.26E+03	0.086
CWOA4	5.56E−39	1.74E−38	0.386	CWOA4	−8.58E+03	1.64E+03	0.017
CWOA5	1.04E−40	1.19E−40	0.037	CWOA5	−1.11E+04	1.10E+03	0.386
CWOA6	1.17E+03	2.10E+02	0.005	CWOA6	−5.59E+03	1.30E+02	0.005
CWOA7	1.17E+03	3.80E+02	0.005	CWOA7	−7.21E+03	2.38E+03	0.008
CWOA8	1.15E+03	1.59E+02	0.005	CWOA8	−8.45E+03	1.46E+03	0.011
CWOA9	4.55E−41	6.96E−41	0.203	CWOA9	−8.46E+03	2.15E+03	0.013
CWOA10	1.67E−40	3.95E−40	0.386	CWOA10	−9.88E+03	1.90E+03	0.028

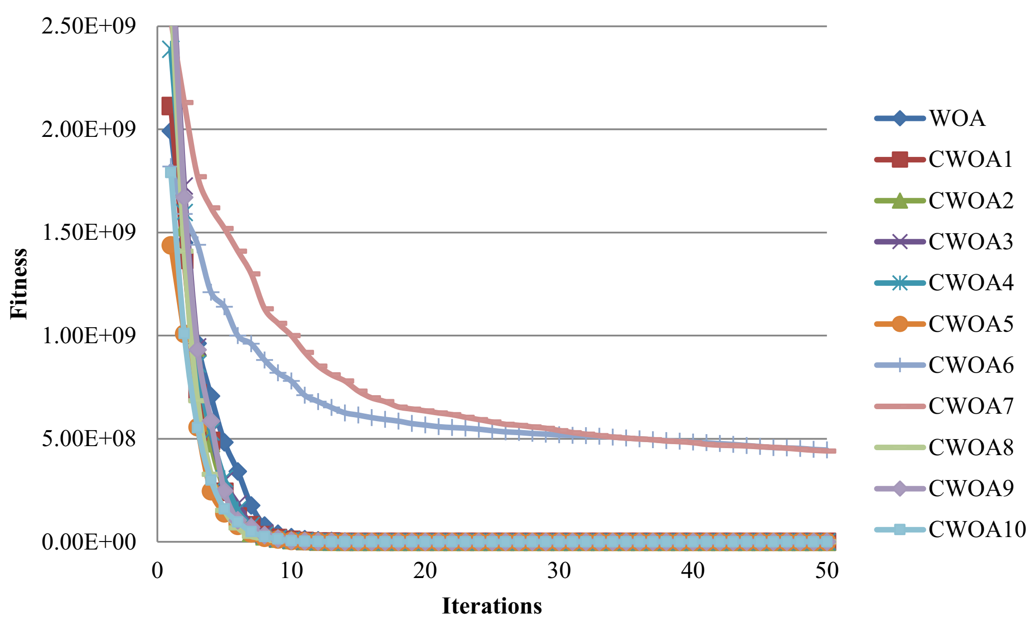

Fig. 2. Performance comparison on the F01 Sphere function.

been done. The line graphs of convergence of various benchmark functions using CWOA algorithm have been shown in Figs. 2–6, which helps to analyze the convergence rate of the algorithm more evidently. To clearly notice and analyze the convergence curves of CWOA on various chaotic maps the graphs have been plotted on 50 iterations. Table 3 shows that on average CWOA8 (Tent map) performs better than other methods on nine of the benchmarks when searching for function minimum. CWOA2, CWOA3 and CWOA7 (Cubic, Sine, and Iterative respectively) are the second best maps performing best on seven out of twenty benchmarks. CWOA1, CWOA4, CWOA5, and CWOA10 (Logistic, Sinusoidal, Singer and Gauss/mouse respectively) are the third most effective and have shown the best performance on six benchmarks. The values shown in these figures are the average function optimum achieved from thirty runs. Here, all the values are true function values.

Fig. 2 shows the values obtained by the ten chaotic maps on the F01 Sphere function, which is also known as *de jong's* function and has a single global value $F01_{min} = 0$, therefore it is easy to solve. From Fig. 2 CWOA9 (Piecewise map) has the fastest convergence rate towards the global solution and overtakes all other methods. CWOA6 and CWOA7 (Circle and Iterative maps respectively) fail to find the global value within the maximum number of iterations.

Fig. 3 displays the function values for the F03 Cigar function. CWOA8 (Tent map) shows the fastest convergence rate and overtakes all other methods. It can also be seen that only CWOA6 and CWOA7 (Circle and Iterative respectively) are inferior to CWOA8 (Tent map). Otherwise, all other maps are very close to CWOA8 in showing a very good convergence rate.

Fig. 4 illustrates the values achieved for the ten methods when using the F08 Greiwank function. F08 has a strange property, as it

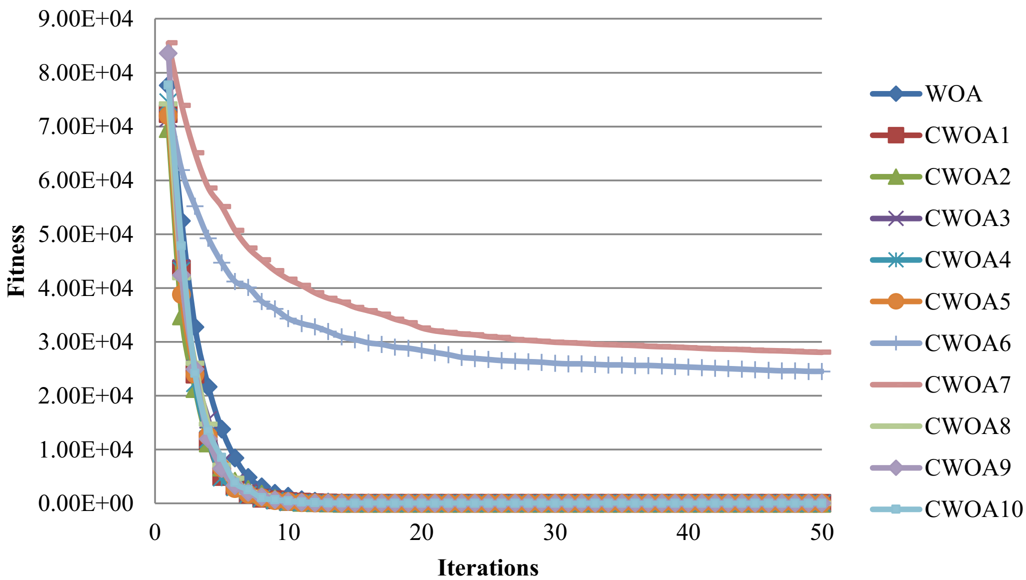

Fig. 3. Performance comparison on the F03Cigar function.

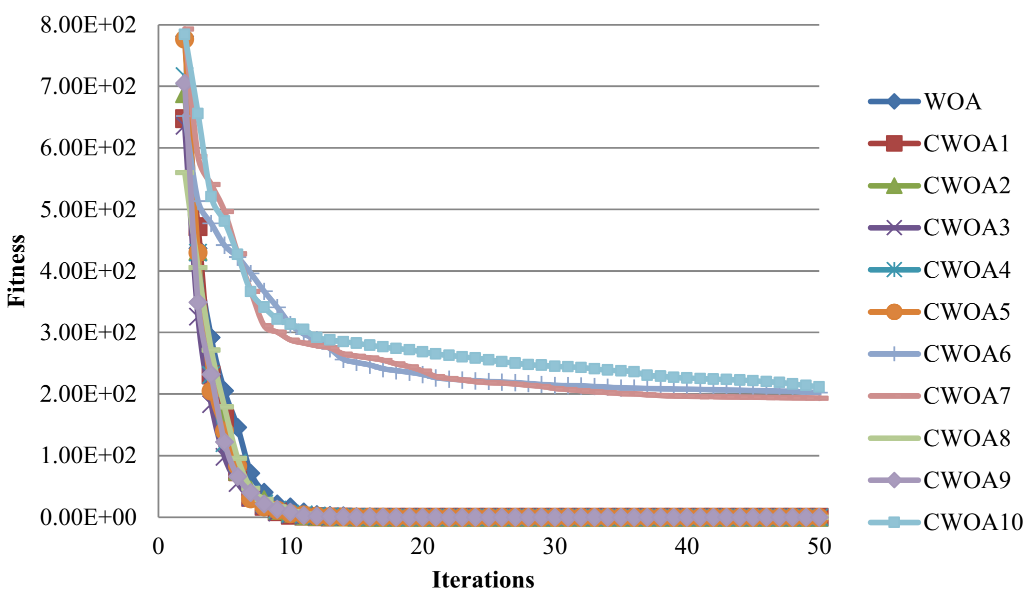

Fig. 4. Performance comparison on the F08 Greiwank function.

is much easier to solve for higher dimensions than lower dimensions (Liang, Qin, Suganthan, & Baskar, 2006). From Fig. 4 all the maps have shown the fastest convergence rate towards the global optimum than WOA except CWOA6, CWOA7, and CWOA10 (Circle, Iterative and Gauss/mouse maps respectively).

Fig. 5 shows the functions values for the F13 Schaffer function, which is a unimodal function. From Fig. 5, CWOA6 (Circle map) performs better than other nine methods while CWOA7 (Iterative map) performs second best in this function.

Fig. 6 reveals the function values for the F18 Schwefel 2.21 function. At first glimpse, it is obvious that CWOA8 has the fastest convergence rate towards the global solution. CWOA8 (Tent map) reaches the optimal solution significantly earlier than other methods. From Fig. 6, it is also illustrated that CWOA9 and CWOA10 (Piecewise and Gauss/mouse maps respectively) are only inferior to CWOA8 (Tent map) and perform second best in this unimodal function.

Considering the results shown in Figs. 2–6, it can be concluded that CWOA has superior performance than WOA. Further, CWOA8 (Tent map) have provided superior results on nine benchmark functions as compared to WOA.

5.3. Statistical testing

To evaluate the performance of meta-heuristic algorithms, statistical tests should be conducted (Derrac, García, Molina, & Herrera, 2011). Specifically, it is not adequate to compare algorithms based on the mean and standard deviation values (García, Molina, Lozano, & Herrera, 2009), and a statistical test is necessary to prove that a proposed new algorithm presents a significant improvement compared to other algorithms. In order to judge whether the results of the algorithms differ from each other in a statistically significant way, a nonparametric statistical test, Wilcoxon's rank-sum test (Wilcoxon, 1945) is carried out at 5%

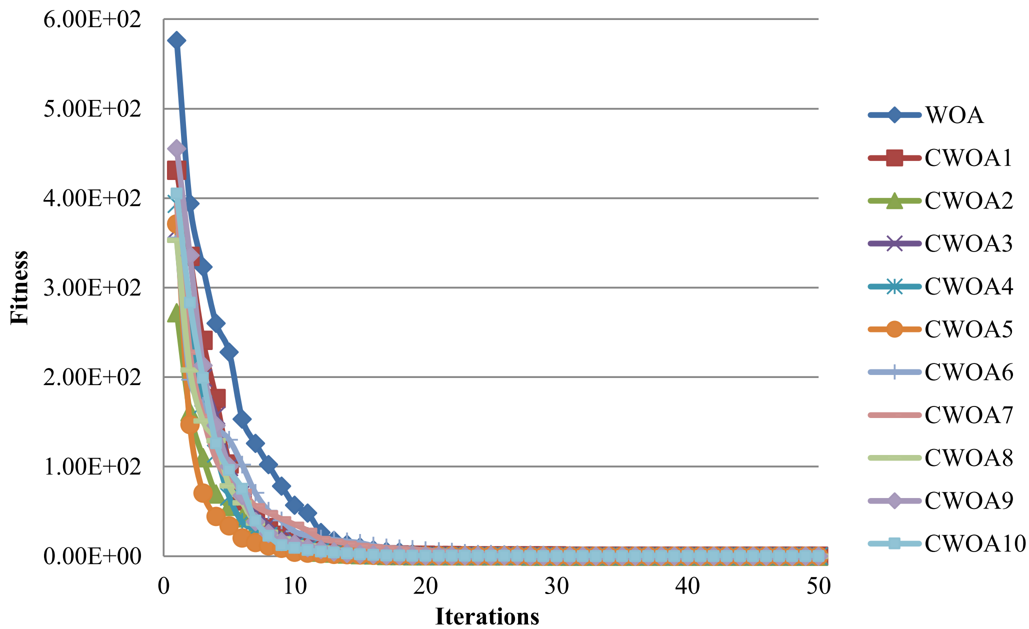

Fig. 5. Performance comparison on the F13 Schaffer function.

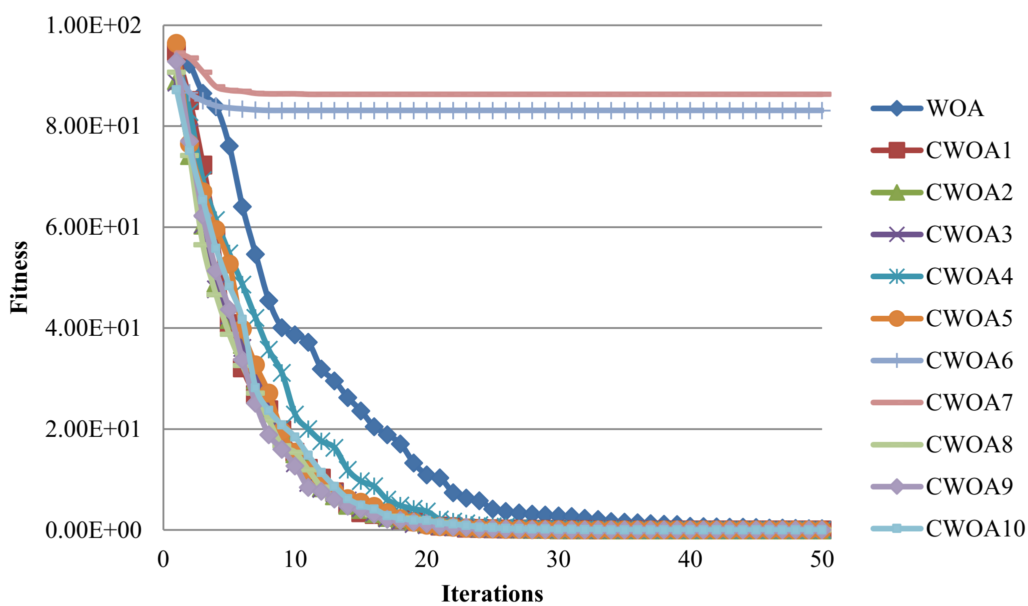

Fig. 6. Performance comparison on the F18 Schwefel 2.21 function.

significance level. The average (mean) and standard deviation (SD) of the best solutions obtained in the last iteration are reflected in Table 3. The p values calculated in the Wilcoxon's rank-sum are given in the results as well. In the tables, N/A indicates "not applicable", meaning that the corresponding algorithm could not be compared with itself in the rank-sum test. Generally, it is considered that p values <0.05 can be considered as sufficient indication against the null hypothesis. Note that the best results are highlighted in bold face and p values <0.05 are underlined. Generally speaking, the results of the chaotic maps on all the benchmark functions follow the order of Tent < Cubic ≈ Since ≈ Piecewise < Logistic≈ Singer ≈ Sinusoidal ≈ Gauss/mouse < Circle ≈ Iterative. Note that the '≈' sign signifies an approximately equal number of successful results of different chaotic maps in many benchmark functions. This comparison shows that the Tent map shows the best (minimum) results, whereas the Circle and the Iterative maps provide the worst (maximum) results. The underlying reason behind the better performance of CWOA using Tent chaotic map is that it provides better exploration and local optima avoidance capability. In other words, the Tentmap bring different patterns of search behavior for whales which results in showing higher exploration capability. The results of convergence curves also prove that the superior exploration of the Tent map does not have a negative impact on the exploitation. To sum up, the results show that the Tent chaotic map shows very effective results and can successfully enhance the convergence rate of CWOA.

6. Conclusions and future scope

In the presented paper chaos theory and whale optimization algorithm (WOA) are hybridized in order to design an improved chaotic whale optimization algorithm (CWOA). To adjust the key parameter, *p*, of WOA, a wide variety of chaotic maps has been utilized. Twenty benchmark functions dividing into multimodal and unimodal problems have been employed in order to compare and verify the performance of CWOAs. Generally speaking, the results proved that chaotic maps are able to significantly improve the performance of WOA. Among all the chaotic maps, the Tent map has considerably enhanced the performance of WOA. The chaos induced by the chaotic maps in the search space is the main reason behind the superior performance of CWOA. The chaos helps the controlling parameter to find the optimal solution more quickly and thus refine the convergence rate of the algorithm. For future work, it would be interesting to employ CWOA algorithm for solving real-world engineering problems.

Conflict of interest

We declare that we have no conflicts of interest in the authorship or publication of this research article.

References

Alatas, B. (2010a). Chaotic harmony search algorithms. *Applied Mathematics and Computation, 216*(9), 2687–2699.

Alatas, B. (2010b). Chaotic bee colony algorithms for global numerical optimization. *Expert Systems with Applications, 37*(8), 5682–5687.

Alba, E., & Dorronsoro, B. (2005). The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. *IEEE Transactions on Evolutionary Computation, 9*(2), 126–142.

Aljarah, I., Faris, H., & Mirjalili, S. (2016). Optimizing connection weights in neural networks using the whale optimization algorithm. *Soft Computing*, 1–15.

Arora, S., & Singh, S. (2015). Butterfly algorithm with Lèvy Flights for global optimization. In *2015 International conference on signal processing, computing and control (ISPCC)* (pp. 220–224). IEEE.

Arora, S., & Singh, S. (2017a). An improved butterfly optimization algorithm with chaos. *Journal of Intelligent & Fuzzy Systems, 32*(1), 1079–1088.

Arora, S., & Singh, S. (2017b). Node localization in wireless sensor networks using butterfly optimization algorithm. *Arabian Journal for Science and Engineering, 42*(8), 3325–3335.

Coello, C. A. C. (2000). Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry, 41*(2), 113–127.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation, 1*(1), 3–18.

Digalakis, J. G., & Margaritis, K. G. (2001). On benchmarking functions for genetic algorithms. *International Journal of Computer Mathematics, 77*(4), 481–506.

Dorigo, M., & Di Caro, G. (1999). Ant colony optimization: A new meta-heuristic. In *CEC 99. Proceedings of the 1999 congress on evolutionary computation, 1999* (Vol. 2, pp. 1470–1477). IEEE.

Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation, 1*(1), 53–66.

Dos Santos Coelho, L., & Mariani, V. C. (2008). Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization. *Expert Systems with Applications, 34*(3), 1905–1913.

Eberhart, R., & Kennedy, J. (1995). Particle swarm optimization. In *Proceedings of IEEE international conference on neural networks, 1995* (pp. 1942–1948).

Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: A new bio-inspired optimization algorithm. *Communications in Nonlinear Science and Numerical Simulation, 17*(12), 4831–4845.

Gandomi, A. H., & Yang, X. S. (2014). Chaotic bat algorithm. *Journal of Computational Science, 5*(2), 224–232.

Gandomi, A. H., Yang, X. S., Alavi, A. H., & Talatahari, S. (2013). Bat algorithm for constrained optimization tasks. *Neural Computing and Applications, 22*(6), 1239–1255.

Gandomi, A. H., Yang, X. S., Talatahari, S., & Alavi, A. H. (2013). Firefly algorithm with chaos. *Communications in Nonlinear Science and Numerical Simulation, 18*(1), 89–98.

Gao, X. Z., Wang, X., Ovaska, S. J., & Xu, H. (2010). A modified harmony search method in constrained optimization. *International Journal of Innovative Computing, Information and Control, 6*(9), 4235–4247.

García, S., Molina, D., Lozano, M., & Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 special session on real parameter optimization. *Journal of Heuristics, 15*(6), 617–644.

He, D., He, C., Jiang, L. G., Zhu, H. W., & Hu, G. R. (2001). Chaotic characteristics of a one-dimensional iterative map with infinite collapses. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 48*(7), 900–906.

Jadhav, A. N., & Gomathi, N. (2017). WGC: Hybridization of exponential grey wolf optimizer with whale optimization for data clustering. *Alexandria Engineering Journal*.

Kaveh, A. (2017). Sizing optimization of skeletal structures using the enhanced whale optimization algorithm. In *Applications of metaheuristic optimization algorithms in civil engineering* (pp. 47–69). Springer.

Kellert, S. H. (1994). *In the wake of chaos: Unpredictable order in dynamical systems*. University of Chicago Press.

Kennedy, J. (2011). Particle swarm optimization. In *Encyclopedia of machine learning* (pp. 760–766). US: Springer.

Kohli, M., & Arora, S. (2017). Chaotic grey wolf optimization algorithm for constrained optimization problems. *Journal of Computational Design and Engineering*.

Liang, J. J., Qin, A. K., Suganthan, P. N., & Baskar, S. (2006). Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Transactions on Evolutionary Computation, 10*(3), 281–295.

Li-Jiang, Y., & Tian-Lun, C. (2002). Application of chaos in genetic algorithms. *Communications in Theoretical Physics, 38*(2), 168.

Liu, B., Wang, L., Jin, Y. H., Tang, F., & Huang, D. X. (2005). Improved particle swarm optimization combined with chaos. *Chaos, Solitons & Fractals, 25*(5), 1261–1271.

Mafarja, M. M., & Mirjalili, S. (2017). Hybrid whale optimization algorithm with simulated annealing for feature selection. *Neurocomputing*.

Mirjalili, S. (2015). The ant lion optimizer. *Advances in Engineering Software, 83*, 80–98.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software, 95*, 51–67.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61.

Pecora, L. M., & Carroll, T. L. (1990). Synchronization in chaotic systems. *Physical Review Letters, 64*(8), 821.

Reddy, P. D. P., Reddy, V. V., & Manohar, T. G. (2017). Whale optimization algorithm for optimal sizing of renewable resources for loss reduction in distribution systems. *Renewables: Wind, Water, and Solar, 4*(1), 3.

Saremi, S., Mirjalili, S., & Lewis, A. (2014). Biogeography-based optimisation with chaos. *Neural Computing and Applications, 25*(5), 1077–1097.

Shu-Chuan, C., Pei-Wei, T., & Jeng-Shyang, P. (2006). Cat swarm optimization'. In *9th Pacific Rim international conference on artificial intelligence, LNAI 4099* (pp. 854–858).

Simon, D. (2008). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation, 12*(6), 702–713.

Sivanandam, S. N., & Deepa, S. N. (2007). *Principles of soft computing*. John Wiley & Sons.

Tsai, P. W., Zhang, J., Zhang, S., Istanda, V., Liao, L. C., & Pan, J. S. (2015). Improving swarm intelligence accuracy with cosine functions for evolved bat algorithm. *Journal of Information Hiding and Multimedia Signal Processing, 6*, 1194–1202.

Wang, G. G., Guo, L., Gandomi, A. H., Hao, G. S., & Wang, H. (2014). Chaotic krill herd algorithm. *Information Sciences, 274*, 17–34.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin, 1*(6), 80–83.

Yang, X. S. (2010). *Nature-inspired metaheuristic algorithms*. Luniver Press.

Yang, X. S. (2010). Firefly algorithm, Levy flights and global optimization. In *Research and development in intelligent systems XXVI* (pp. 209–218). London: Springer.

Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization (NICSO 2010)* (pp. 65–74).

Yang, D., Li, G., & Cheng, G. (2007). On the efficiency of chaos optimization algorithms for global optimization. *Chaos, Solitons & Fractals, 34*(4), 1366–1375.

Yao, X., Liu, Y., & Lin, G. (1999). Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation, 3*(2), 82–102.