ขั้นตอนวิธีการเพิ่มประสิทธิภาพซาตินโบเวอร์เบิร์ดกับความอลวน

Satin Bowerbird Optimization Algorithm with Chaos

ธนัชพงษ์ วังคำหาญ

Tanachapong Wangkhamhan

Received: 11 April 2019 ; Revised: 14 June 2019 ; Accepted: 15 July 2019

บทคัดย่อ

การหาค่าที่เหมาะสมที่สุดของซาตินโบเวอร์เบิร์ด เป็นขั้นตอนวิธีการหาค่าเหมาะที่พัฒนาโดยขั้นตอนวิธีเมตาฮิวริสติกซึ่งถูกพัฒนาเมื่อเร็ว ๆ นี้ โดยได้รับแรงบันดาลใจจากนกซาตินโบเวอร์ ที่อาศัยอยู่ในป่าฝนของออสเตรเลียและแหล่งอาศัยอื่น ๆ เช่นเดียวกับขั้นตอนวิธีเมตาฮิวริสติกอื่น ๆ ปัญหาหลักที่ขั้นตอนวิธีซาตินโบเวอร์เบิร์ดเผชิญอยู่ซึ่งได้รับการพิสูจน์อย่างชัดเจน การติดอยู่ในค่าคำตอบที่ดีสุดเฉพาะที่อย่างง่ายดาย มีความแม่นยำต่ำ และความเร็วในการลู่เข้าแก้ปัญหาการหาค่าเหมาะสมที่ช้า ดังนั้นในความพยายามที่จะเพิ่มความเร็วในการลู่เข้าแก้ปัญหาการหาค่าเหมาะสมที่แท้จริง และได้รับประสิทธิภาพที่ดีขึ้น บทความนี้จะนำเสนอทฤษฎีความอลวนในกระบวนการเพิ่มประสิทธิภาพขั้นตอนวิธีซาตินโบเวอร์เบิร์ด ตัวแปรความวุ่นวายในแมปจะถูกนำมาพิจารณาโดยการนำเสนอวิธีความอลวนกับขั้นตอนวิธีซาตินโบเวอร์เบิร์ด เพื่อที่จะแทนที่ตัวแปรหลัก (α) ซึ่งช่วยในการควบคุมทั้งการสำรวจพื้นที่และการนำไปใช้ประโยชน์ของขั้นตอนวิธีซาตินโบเวอร์เบิร์ด วิธีการที่นำเสนอจะถูกเปรียบเทียบในการทดสอบปัญหา CEC2014 ผลลัพธ์เชิงตัวเลขแสดงให้เห็นถึงขั้นตอนวิธีที่ถูกปรับปรุงด้วยแมปความอลวน โดยเฉพาะในเต้นท์แมป สามารถปรับปรุงประสิทธิภาพของขั้นตอนวิธีซาตินโบเวอร์เบิร์ดดั้งเดิมได้ ซึ่งมีประสิทธิภาพที่ดีที่สุด

คำสำคัญ: การเพิ่มประสิทธิภาพซาตินโบเวอร์เบิร์ด ขั้นตอนวิธีเมตาฮิวริสติก ทฤษฎีความอลวน แมปความอลวน

Abstract

The Satin Bowerbird Optimization (SBO) is a recently developed meta-heuristic optimization algorithm which was inspired by the Satin Bowerbirds living in Australia's rainforests and other mesic habitats.Like other meta-heuristic algorithms, the main problem faced by the SBO is that it has been empirically demonstrated to become easily trapped into local optimal solutions, creating low precision and slow convergence speeds. Therefore, in an effort to enhance global convergence speeds and in order to obtain a better performance, this paper introduces the chaos theory into the SBO optimization process. Various chaotic maps were considered in the proposed Chaotic-SBO (CSBO) method in order to replace the main parameter's greatest step size (), which assists in controlling both exploration and exploitation. The proposed CSBO methods are benchmarked within the CEC2014 test problems. The numerical results show that these novel algorithms improve the chaotic maps, especially in the Tent map, as well as improving the performance of the original Satin Bowerbird Optimization Algorithm. Specifically, the CSBO10(Tent) achieved the best performance.

Keywords: Satin Bowerbird Optimization, Meta-heuristic algorithm, chaos theory, chaotic map.

อาจารย์, คณะบริหารศาสตร์ มหาวิทยาลัยกาฬสินธุ์ อำเภอเมืองกาฬสินธุ์ จังหวัดกาฬสินธุ์ 46000

Lecturer, Faculty of Management Science, Kalasin University, Kalasin 46000, Thailand.

Corresponding author; Tanachapong Wangkhamhan, Faculty of Management Science, Kalasin University, Kalasin 46000, Thailand. tanachapong.w@hotmail.com.

1. Introduction

The optimization problem aims to search for a maximum or minimum of an objective function value in widely varied local optima, under highly complex constraints, and in a reasonable amount of time[1]. Consequently, chaotic sequences generated by means of chaotic maps have been used in the development of global optimization techniques. The first introduction of chaos into the optimization challenge was the Chaos Optimization Algorithm (COA) in 1963 by E.N. Lorenz[2].The COA represents the bounded, unstable, dynamic behavior that exhibits sensitive dependence on its initial conditions[3]. Uniquely characteristic of chaotic behavior, the COA carries out global exploration searchs at higher speeds than stochastic ergodic searches, which are dependent on probabilities[4]. Chaos is a characteristic of several nonlinear systems as motion distributes within a specific range since it possesses degrees of uncertainty, ergodicity, and stochasticity. Many researchers therefore use the characteristics of chaotic ergodicity to solve for the global optimal solution of complex nonlinear multi-peak problems, by weakening the randomness or constant parameters of the metaheuristic optimization algorithm[5]. As a result, most current works are devoted to the improvement of global optimization algorithms to tackle the abovementioned shortcomings. Further interest has been developed in the field of hybrid algorithms, especially in typical and emerging heuristic optimization algorithms; such as the harmony search algorithm [3], migration and merging operation[4], cuckoo search optimization algorithms[5], bee colony algorithm[6], great deluge algorithm[7], imperialist competitive algorithm[8], particle swarm optimization[9,10], firefly algorithm[11], gravitational search algorithm[12], grey wolf optimization algorithm[13], whale optimization algorithm[14], crow search algorithm[15,16], league championship algorithm[17], salp swarm algorithm[18], and the krill herd algorithm[19]; all of which were hybridized with the COA. Various simulation results and applications in each of these references have proven the solution diversity and global optimization capacity of each chaos based optimization algorithm.

The standard Satin Bowerbird Optimizer (SBO) was first proposed by S. H. Samareh Moosavi and V. Khatibi Bardsiri[20] in 2017 to optimize ANFIS for the purpose of software development effort estimation. Its algorithm was bio-inspired by Satin Bowerbirds living in the rainforests and mesic habitats of Australia. Through the breeding principle of male-attracting-female, the male bowerbird attracts the female with the construction of a specialized bower. The SBO technique, which is population-based on a stochastic optimization algorithm[21], is very robust, straightforward, and efficient. Details of the original SBO and the literature related to its applications are presented in Sections 4 and 6.

Our principle concern we faced in our research was to introduce the Chaotic Satin Bowerbird Optimizer (CSBO) based methods in ways in which different chaotic systems are used to replace the critical parameters of the SBO. Through this method, we intended to enhance the global searching ability of the SBO, and increase its ability to stick on a local solution. The simulation results demonstrated the improved performance of the CSBO with the application of the deterministic chaotic signals, as opposed to the constant parameters of the SBO.

The remainder of this paper is organized in seven sections. Section 2 briefly describes the original SBO algorithm; Section 3 describes the chaotic maps; the proposed CSBO approach is detailed in Section 4; and the benchmark functions and parameter studies used to test the performance of the CSBO are found in Section 5. The experimental results and discussions are presented in Section 6, and the conclusions and future scope of our research are presented in Section 7.

2. The original SBO algorithm

The SBO algorithm[20,21] is a population-based algorithm which estimates the global optimum for a given optimization problem. This process starts by creating a population of random uniform distribution, through the consideration of both the lower and upper limit parameters. After that, each position is defined as a dimensional vector of the parameters, which must be optimized.The

probability of such defines the attractiveness of the bower. A female satin bower bird selects a bower (nest) based on its probability, and is able to calculate the probability of each population member through Eqs. (1) and (2), below.

$$\mathrm{Prob}_i = \frac{\mathrm{fit}_i}{\sum_{n=1}^{\mathrm{NB}} \mathrm{fit}_n} \ , \tag{1}$$

$$\mathrm{fit}_i = \begin{cases} \frac{1}{1+f(x_i)}, f(x_i) \geq 0 \\ 1 + |f(x_i)|, f(x_i) < 0 \end{cases} , \tag{2}$$

where NB is the population size of the bower, is the fitness value of the *i*th solution, and $f(x_i)$ is the fitness value of *i*th bower. To find the position of the best bower, the SBO algorithm utilizes the concept of Elitism, which allows the best solution to be preserved at each stage of the optimization process. The SBO algorithm uses the concept of birds building their nests using their natural instincts. The male satin bower bird likes all other birds in the mating season, and uses his natural instincts to build his bower and decorate it. We may infer that the male bowers rely upon their experience to influence their creative decisions in building their bower; therefore, more experienced birds will build more attractive bowers (improving their fitness) than others. In this work, the best built bower (best position) is intended as an elite iteration. Since the elite position has the highest fitness, it should be able to influence the other positions. The changes of each new bower, representing a new position determined by the position of the best fit bower (position), are calculated according to Eq. (3).

$$x_{ik}^{new} = x_{ik}^{old} + \lambda_k \left(\left(\frac{x_{jk} + x_{elite,k}}{2} \right) - x_{ik}^{old} \right) , \tag{3}$$

where x_i is the *i*th solution vector (bower), x_j is determined as the target solution among all solutions in the current iteration, j is calculated by the roulette wheel procedure, and x_{ik} is the *k*th member of this dimensions. x_{elite} indicates the elite position (the best fitness value in the current iteration).

$$\lambda_k = \frac{\alpha}{1 + p_j} . \tag{4}$$

In Eq. (4), determines the attraction power in the goal bower, shown at intervals of , where is the greatest step size (constant) and is the probability obtained by Eq. (1) using the goal bower, at intervals of .In the mutation process, which occurs at the completion of each iteration of the SBO, random changes are applied with a certain probability. Random changes are then applied to , again, with a certain probability.The normal distribution within the mutation process is employed through the average of and the variance of , as seen in Eqs. (5)-(7).

$$x_{ik}^{new} \sim N(x_{ik}^{old}, \sigma^2) \ , \tag{5}$$

$$N(x_{ik}^{old}, \sigma^2) = x_{ik}^{old} + (\sigma^* N(0,1)) \ , \tag{6}$$

$$\sigma = Z^* (\mathrm{var}_{\max} - \mathrm{var}_{\min}) \ , \tag{7}$$

where σ is a proportion of space width, and and are the lower and upper bounds assigned to the variables, respectively. The value of the Z parameter is the percent of the difference between the lower and upper limit, which is variable. In the last of stage of each iteration, the newly formed population and the initial population are evaluated, and all populations are combined and sorted by their fitness values. A new population is then created according to the previously defined number, while the others are rejected. The basic steps of the SBO are shown in Algorithm 1.

Algorithm 1 The satin bowerbird optimization algorithm[20].

1: Initialize the population size of bowers (NB), greatest step size (α), mutation probability (P), percentage of the difference between the upper and lower limits (Z) and proportion of space width (σ) calculated by Eq. (7)
2: Generate the population (bower)
3: Evaluate the fitness value of bowers. Let the initialization also be the best bower and assume it as elite
4: **While** (the end criterion is not satisfied) **Do**
5: Calculate the probability of bowers using Eqs. (1) and (2)
6: **For** $i = 1$ to all bower **Do**
7: **For** $j = 1$ to all element of bower **Do**
8: Select one bower randomly using roulette wheel selection
9: Calculate step size (λ_k) using Eq. (4)
10: Update the position of bower using Eqs. (3) and (6)
11: **End for**
12: Evaluate the fitness value of bower.
13: **End for**
14: Sorted bower and by the fitness values
15: Find the current global best
16: **End while**
17: **Output** the best fitness value of bower

3. Chaotic maps for the SBO

This section presents the descriptions and equations of ten discrete chaotic maps used within the experiments of the SBO algorithm.

Chaos, as a kind of dynamic behavior within a nonlinear chaotic time series, has raised enormous interest in fields such as scientific applications and engineering systems[22]; which have included numerical simulation, chaos control, synchro-nization, pattern recognition, optimization theory, and additional nonlinear sciences. In random–based optimization algorithms, methods employing *chaotic* variables rather than *random* variables are referred to as chaotic optimization algorithms (COA)[11]. Due to the non-repetition of chaotic behavior, the algorithm is capable of carrying out overall searches at higher speeds than stochastic searches, which are dependent upon their probabilities[23]. One–dimensional, non–invertible maps are the simplest systems capable of generating the desired chaotic motion[24].

To fulfill this matter within our study, one–dimensional, non–invertible maps are utilized to generate chaotic sets. In Table 1, we review some of the better known one–dimensional maps. Within the ten different chaotic maps presented in this table; k indicates the index of the chaotic sequence, and x_k represents the *k*th number in the chaotic sequence. Figure 1a - 1j outlines the dynamics of the ten different chaotic maps.

Table 1 The ten chaotic maps[13,14,16,19].

No.	Name	Definition
CSBO1	Chebyshev map	$x_{k+1} = \cos(k \cos^{-1}(x_k))$
CSBO2	Circle map	$x_{k+1} = x_k + b - (a - 2\pi)\sin(2\pi x_k) \bmod(1)$ With a = 0.5 and b = 0.2, it generates a chaotic sequence in (0, 1)
CSBO3	Gauss/Mouse map	$x_{k+1} = \begin{cases} 0 & x_k = 0 \\ 1/k \bmod(1) & \text{otherwise} \end{cases}$, $1/x_k \bmod(1) = \frac{1}{x_k} - \left[\frac{1}{x_k}\right]$
CSBO4	Iterative map	$x_{k+1} = \sin\left(\frac{a\pi}{x_k}\right)$, $a \in (0, 1)$
CSBO5	Logistic map	$x_{k+1} = a x_k (1 - x_k)$
CSBO6	Piecewise map	$x_{k+1} = \begin{cases} \frac{x_k}{P} & 0 \le x_k < P \\ \frac{x_k - P}{0.5 - P} & P \le x_k < \frac{1}{2} \\ \frac{1 - P - x_k}{0.5 - P} & \frac{1}{2} \le x_k < 1 - P \\ \frac{1 - x_k}{P} & 1 - P \le x_k < 1 \end{cases}$
CSBO7	Sine map	$x_{k+1} = \frac{a}{4}\sin(\pi x_k)$, $0 < a \le 4$
CSBO8	Singer map	$x_{k+1} = \mu(7.89 x_k - 23.31 x_k^2 + 28.75 x_k^3 - 13.3028 .75 x_k^4)$
CSBO9	Sinusoidal map	$x_{k+1} = a x_k^2 \sin(\pi x_k)$, a = 2.3 and x_0 = 0.7
CSBO10	Tent map	$x_{k+1} = \begin{cases} \frac{x_k}{0.7} & x_k < 0.7 \\ \frac{10}{3}(1 - x_k) & x_k \ge 0.7 \end{cases}$

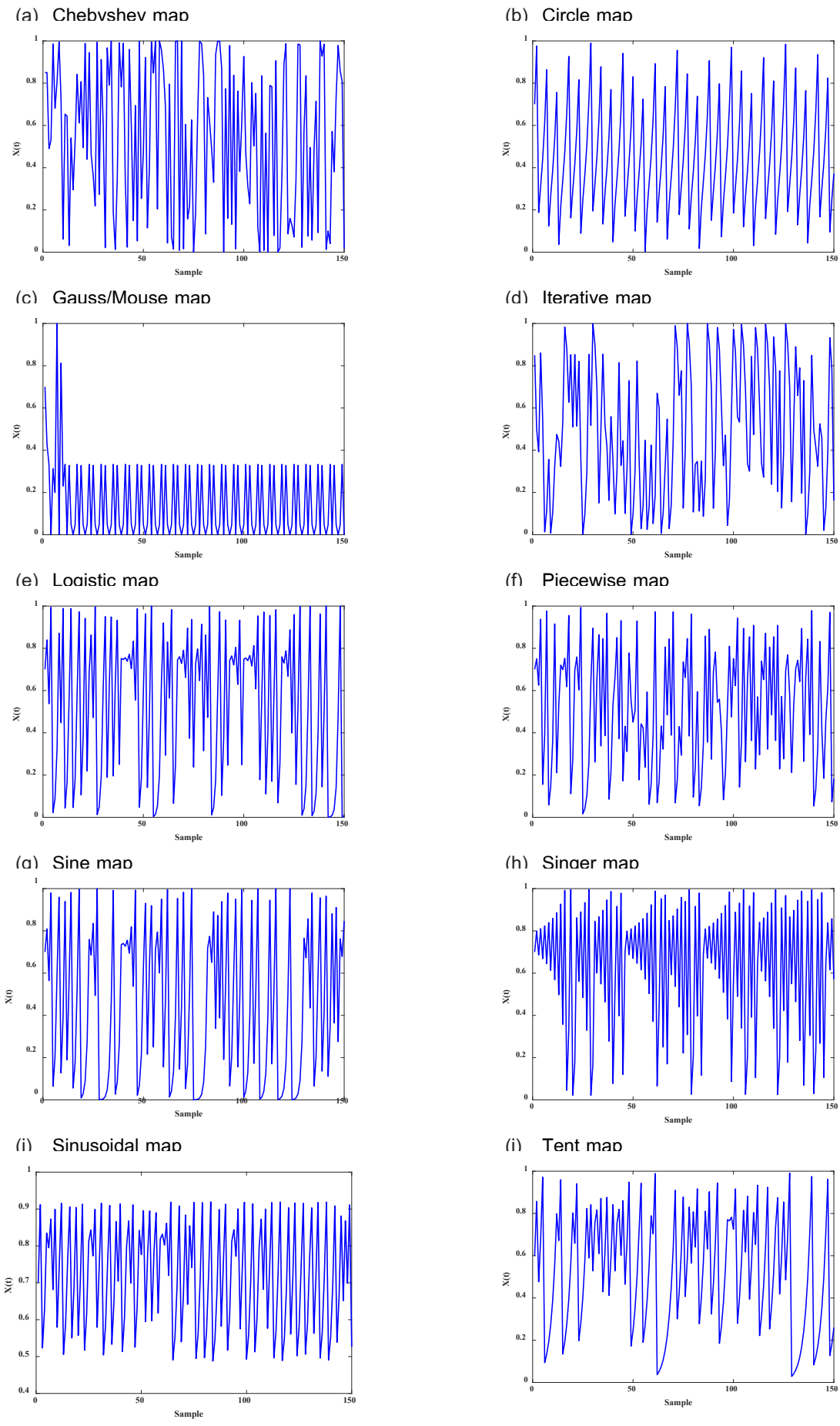


Figure 1 The dynamics of the chaotic maps $\chi^t \in (0, 1)$.

4. The proposed CSBO approach

This section presents a novel Chaotic with Satin Bowerbird Optimization algorithm called the CSBO, which replaces the main parameter and embeds chaos into the existing SBO.While SBOs possess good convergence rates, they still cannot perform sufficiently in finding the global optima, which in turn affects the convergence rate of the algorithm.In order to reduce this effect and to improve its efficiency, the concept of chaos has been introduced into the SBO algorithm.

Chaotic maps are imbedded into the SBO to improve the algorithm's solution quality. One of the main parameters of the SBO is the greatest step size (α), which remains a constant parameter. Here, this value (α) is replaced with ten different chaotic maps in an attempt to improve the performance of the SBO. To implement the maps, all maps are normalized between 0 and 1. Furthermore, the parameter of α, determined by Eq. (4), is modified by the ten different chaotic maps through the following equation, Eq. (8).

$$\lambda_k = \begin{cases} \dfrac{c^{t+1}}{1+p_j} & \text{mod}(t, NB-1) = 0, \\ \dfrac{c^{t+1}}{1+p_j} & \text{otherwise} \end{cases} . \quad (8)$$

Where c^{t+1} represents the different chaotic variables, is current iteration $(t = 1, 2, 3, \ldots m)$, m represents the maximum iteration number, and is the number of bowers. Eq. (8) produces a design point x_{ik}^{new}, from Eq. (3), which uses the different chaotic variables derived from the chaotic maps, with different initial values.

The pseudo-code of the CSBO algorithm is presented in Algorithm 2. In the first step, the bower population within the search space is initialized randomly. After which, the parameters of the CSBO algorithm involved in controlling the exploration and exploitation mechanisms, specifically the NB, P, Z, σ, and α, are initialized similarly to the SBO. In the second step, the fitness function values of all bowers are initialized in the search space, and evaluated using the various standard benchmark functions. The lower fitness value is assumed to be the elite (the best fitness function value). The chaotic number of the chaotic map is initialized to adjust the parameter of the SBO. In the third step, the CSBO algorithm runs sequentially, in which all bowers will update their positions, resulting in the first position as the optimal solution. The value of parameter is also updated along with the course of each iteration through Eq. (8), where $mod(t, NB-1) = 0$, t is the current iteration, and is the population size (Algorithm 2, line 19). In the finally step, at the end of the last iteration, the best search agent will be considered as the most optimal solution by the CSBO algorithm.

5. Implementation

5.1 Benchmark numerical experiments

Ten different chaotic SBO variants of this comparative study are presented in the 30 benchmark problems taken from the special session and competition on single objective real-parameter numerical optimization held under the IEEE Congress on Evolutionary Computation (CEC) 2014[25]; available on http://www3.ntu.edu.sg/home/EPNSugan/index_files/CEC2014/CEC2014.htm. The formulation of these benchmark functions are presented in Table 2.

Algorithm 2 The CSBO algorithm.

1: Initialize the population size of bowers (NB), greatest step size (α), mutation probability (P), percentage of the difference between the upper and lower limits (Z), proportion of space width (σ) and $NFEs = 0$
2: Generate the population $X_i = (i = 1, 2, 3, \dots, N)$ of N bowers
3: **For** i = 1 to N **Do**
4: Evaluate the fitness value of all bowers $f(X_i)$
5: The best bower (X_{best}) and assume it as elite
6: $NFEs = NFEs + NFEs$ that is consumed by bower
7: **End for**
// **The stage of CSBO**
8: Initial iterations $t = 1$
9: Generate the chaotic sequences $c_1^t \in (0, 1)$, the description in Section 3 and Table 1 ◁ (1)
10: **While** ($NFEs \geq max\_NFEs$) **Do**
11: **For** $k = 1$ to N **Do**
12: Calculate the probability (P) of bowers using Eqs. (1) and (2)
13: **End for**
14: **//Generate a new bower** $(\boldsymbol{X_i^{t+1}})$
15: **For** $i = 1$ to N **Do**
16: **For** $k = 1$ to D (all element (D) of bower) **Do**
17: Select one bower (X_j^t), where (X_j^t) is random using roulette wheel selection
18: **//Calculate step size** $(\boldsymbol{\lambda_k})$
19: **If** $mod(t, N-1) = 0$ **Then** ◁ (2)
20: Calculate step size (λ_k) using Eq. (8) ◁ (3)
21: **Else**
22: Calculate step size (λ_k) using Eq. (4)
23: **End if**
24: Update the position of bower (X_i^{t+1}) using Eq. (3)
25: **//Mutation**
26: **If** $rand \leq P$ **Then**
27: Update the position of bower (X_i^{t+1}) using Eq. (6)
28: **End if**
29: **End for**
30: Evaluate the fitness value of bower $f(X_i^{t+1})$
31: $NFEs = NFEs + 1$
32: **End for**
33: Sorted bower (X_N) and $f(X_N)$ by the fitness values
34: Update elite (X_{best}) if a bower becomes fitter than the elite
35: $t = t + 1$
36: **End while**
37: **Output** the global best fitness value of bower (X_{best})

Note: The differences between the SBO and CSBO are indicated with lines marked with the symbol ◁ (·).

Table 2 Description of the CEC2014 benchmark functions [26–28].

	Func. #	Function name	Md	Sp	Ro	Sh	Hy	Co	DFVS	DFLB	Opt.
Unimodal	f_1	Rotated High Conditioned Elliptic Function	U	N	Y	N	N	N	N	N	100
	f_2	Rotated Bent Cigar Function	U	N	Y	N	N	N	N	N	200
	f_3	Rotated Discus Function	U	N	Y	N	N	N	N	N	300
Simple Multimodal	f_4	Shifted and Rotated Rosenbrock's Function	M	N	Y	Y	N	N	N	N	400
	f_5	Shifted and Rotated Ackley's Function	M	N	Y	Y	N	N	N	N	500
	f_6	Shifted and Rotated Weierstrass Function	M	N	Y	Y	N	N	N	N	600
	f_7	Shifted and Rotated Griewank's Function	M	N	Y	Y	N	N	N	N	700
	f_8	Shifted Rastrigin's Function	M	Y	N	Y	N	N	N	N	800
	f_9	Shifted and Rotated Rastrigin's Function	M	N	Y	Y	N	N	N	N	900
	f_{10}	Shifted Schwefel's Function	M	Y	N	Y	N	N	N	N	1000
	f_{11}	Shifted and Rotated Schwefel's Function	M	N	Y	Y	N	N	N	N	1100
	f_{12}	Shifted and Rotated Katsuura Function	M	N	Y	Y	N	N	N	N	1200
	f_{13}	Shifted and Rotated HappyCat Function	M	N	Y	Y	N	N	N	N	1300
	f_{14}	Shifted and Rotated HGBat Function	M	N	Y	Y	N	N	N	N	1400
	f_{15}	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	M	N	Y	Y	N	N	N	N	1500
	f_{16}	Shifted and Rotated Expanded Scaffer's F6 Function	M	N	Y	Y	N	N	N	N	1600
Hybrid	f_{17}	Hybrid Function 1 (N = 3)	M/U	N	N	N	Y	N	Y	N	1700
	f_{18}	Hybrid Function 2 (N = 3)	M/U	N	N	N	Y	N	Y	N	1800
	f_{19}	Hybrid Function 3 (N = 4)	M/U	N	N	N	Y	N	Y	N	1900
	f_{20}	Hybrid Function 4 (N = 4)	M/U	N	N	N	Y	N	Y	N	2000
	f_{21}	Hybrid Function 5 (N = 5)	M/U	N	N	N	Y	N	Y	N	2100
	f_{22}	Hybrid Function 6 (N = 5)	M/U	N	N	N	Y	N	Y	N	2200
Composition	f_{23}	Composition Function 1 (N = 5)	M	N	N	N	N	Y	N	Y	2300
	f_{24}	Composition Function 2 (N = 3)	M	N	N	N	N	Y	N	Y	2400
	f_{25}	Composition Function 3 (N = 3)	M	N	N	N	N	Y	N	Y	2500
	f_{26}	Composition Function 4 (N = 5)	M	N	N	N	N	Y	N	Y	2600
	f_{27}	Composition Function 5 (N = 5)	M	N	N	N	N	Y	N	Y	2700
	f_{28}	Composition Function 6 (N = 5)	M	N	N	N	N	Y	N	Y	2800
	f_{29}	Composition Function 7 (N = 3)	M	N	N	N	N	Y	Y	Y	2900
	f_{30}	Composition Function 8 (N = 3)	M	N	N	N	N	Y	Y	Y	3000

Node: Features of the 30 benchmark functions (Note: Md: Modality, Sp: Separable, Ro: Rotated, Sh: Shifted, Hy: Hybrid, Co: Composition, DFVS: Different properties for different variables subcomponents, DFLB: Different properties around different local optima, U: Unimodal, M: Simple Multimodal, N: No, Y: Yes.)

5.2 Initialization and parameter settings

In an effort to produce meaningful statistics, the experiment was carried out over 50 different runs for each setting, with the same initial conditions for all algorithms. Moreover, the performance of the CSBO algorithms was measured using statistical measures, such as mean objective function value and the standard deviation. The population size of the SBO and among the CSBO algorithms was, which we felt would be sufficient for most problems. With a fixed population size at each run, the benchmark function test problems were executed with 30 and 50 dimensions of 50 function evaluations. The maximum number of functions evaluated ($NFEs$) were set at 3.0E+5 and 5.0E+5[29–31], respectively. The percentage of the difference between the upper and lower limits (Z) was considered to be 0.02, the mutation probability (P) was 0.05, the greatest step size (α) was 0.94, and the proportion of space width (σ) was calculated through Eq. (7)[20,21].

6. Experiment results and discussion

In this paper, our experiments are coded in MATLAB R2016a, 64 bit, and run on a desktop computer with an Intel® Core™ i7-6770HQ processor, 8.00GB of RAM, 500GB of HD, and Microsoft Windows 10 Professional 64 bit Operating System. The sets of benchmark test functions were employed to demonstrate the effectiveness of the SBO and CSBOs (denoted as CSBO1 to CSBO10), and the 30 test functions with 30 dimensions (30D) and 50 dimensions (50D), via IEEE CEC2014.

The average objective function values ("Avg.Obj") and standard deviation of the fitness function values ("Std. Dev") of all runs were recorded. The "Avg.Obj" and "Std. Dev" were considered as two performance metrics to assess the performance of the algorithms. Moreover, Wilcoxon's rank sum test[32] at a 5% significance level was used to test the statistical significance between pairwise algorithms.

The performance of the CSBO with different chaotic maps and their results are discussed in Section 6.1. The qualitative analysis is described in Section 6.2, and the analysis ranked by Friedman rank is described in Section 6.3. Also, the statistical testing by Wilcoxon's rank sum test of the experiment results is clearly described in Section 6.4.

6.1. Performance assessment of CSBO with different chaotic maps

Within the experiment results, the CSBOs utilized the Chebyshev, Circle, Gauss/Mouse, Iterative, Logistic, Piecewise, Sine, Singer, Sinusoidal and Tent map, respectively; as shown in Table 1 and Figure1.The CSBOs are capable of significantly improving the solution quality through the use of the chaotic maps. The adjustments of the main parameter are implemented with the various chaotic maps, as seen in Section 4. In Tables 3 and 4, the SBO, CSBO1(Chebyshev), CSBO2(Circle), CSBO3-(Gauss/Mouse), CSBO4(Iterative), CSBO5(Logistic), CSBO6(Piecewise), CSBO7(Sine), CSBO8(Singer), CSBO9(Sinusoidal) and CSBO10(Tent) were proven to successfully enhance the performance of the SBO algorithm. However, compared to the CSBO algorithm, the CSBO3(Gauss/Mouse) algorithm produced less satisfying results. The most favorable results, derived from the Wilcoxon's rank sum test on the 50 independent runs in both 30 and 50 dimensions, were achieved by the CSBO10(Tent) algorithm (Tables 7 and 8). The *p*-Values clearly depict that this supremacy is statistically significant.

The average running times of the SBO and among the CSBO algorithms were collected and are presented at the end of Tables 3 and 4, in which all methods of the CSBO clearly outperformed the SBO algorithm, and in which the CSBO10(Tent) was the fastest of all algorithms.

Table 3 Comparisons of the CSBO with ten different CSBO variants (CSBO1 to 10) through 50 independent runs on 30 test functions with 30 dimensions, taken from IEEE the CEC2014 using 300,000 *NFEs*.

#	Criteria	SBO	CSBO1	CSBO2	CSBO3	CSBO4	CSBO5	CSBO6	CSBO7	CSBO8	CSBO9	CSBO10
f_1	Avg.Obj	2954501.4906	2284003.5525	2218097.3380	2257696.9538	2177756.9027	405202.1121	2109640.1369	2208077.6958	2196663.4872	2146540.2880	**375480.9475**
	Std.Dev.	8.6304E+05	9.2116E+05	9.1281E+05	1.0582E+06	9.4940E+05	2.2039E+05	9.5386E+05	8.6263E+05	8.2349E+05	9.4494E+05	**2.0571E+05**
f_2	Avg.Obj	537627.9478	12085.0254	11823.4205	12169.4099	11973.1865	11196.4560	11805.0292	11878.8561	11636.9373	11990.3100	**11190.3635**
	Std.Dev.	7.8532E+04	6.4691E+03	6.2139E+03	7.3587E+03	6.5943E+03	5.8514E+03	6.1463E+03	6.1872E+03	6.0662E+03	6.2673E+03	**5.4370E+03**
f_3	Avg.Obj	11467.7606	6603.9789	7816.7485	8377.5457	6980.0714	3547.0344	7539.7774	6826.0566	8027.3659	7253.6294	**3090.3038**
	Std.Dev.	6.2529E+03	5.5419E+03	6.9721E+03	6.5701E+03	6.6878E+03	3.3481E+03	5.2453E+03	5.7181E+03	6.7682E+03	6.3969E+03	**2.9297E+03**
f_4	Avg.Obj	536.5799	501.6470	495.4016	488.8676	494.3735	453.9734	498.7470	504.0369	496.4601	500.5301	**452.9273**
	Std.Dev.	2.4539E+01	3.1270E+01	3.5888E+01	3.0808E+01	3.5293E+01	3.2875E+01	3.3522E+01	3.5840E+01	3.2783E+01	3.5856E+01	**3.5539E+01**
f_5	Avg.Obj	520.8515	520.6840	520.7238	520.8052	520.7045	520.6256	520.6345	520.6531	520.7432	520.6680	**520.6225**
	Std.Dev.	7.4616E-02	2.3825E-01	2.2421E-01	6.9573E-02	2.5296E-01	2.7599E-01	2.9140E-01	2.6467E-01	2.2863E-01	2.7333E-01	**2.9740E-01**
f_6	Avg.Obj	628.4728	627.4952	626.7816	627.0822	627.5141	**625.1058**	627.5297	627.1354	627.4688	627.2582	625.3411
	Std.Dev.	1.2209E+00	2.9655E+00	2.8788E+00	2.9168E+00	3.1261E+00	**3.5835E+00**	3.0883E+00	2.7242E+00	3.3775E+00	2.8582E+00	3.5974E+00
f_7	Avg.Obj	700.9321	700.1082	700.1147	700.0970	700.1012	700.0735	700.0834	700.1199	700.1389	700.1075	**700.0696**
	Std.Dev.	3.5790E-02	7.1906E-02	1.1594E-01	3.9728E-02	9.4224E-02	4.4297E-02	3.1915E-02	1.0730E-01	1.7225E-01	8.0747E-02	**2.0174E-02**
f_8	Avg.Obj	942.9053	936.1531	932.4561	936.1521	938.7979	933.1951	937.3471	936.7560	936.0223	936.5516	**932.1518**
	Std.Dev.	2.6164E+01	2.4472E+01	2.2606E+01	2.0077E+01	2.3353E+01	1.9875E+01	2.3125E+01	2.5205E+01	2.0723E+01	1.8947E+01	**2.4637E+01**
f_9	Avg.Obj	1087.7137	1084.5396	1083.9894	1079.8307	1084.7454	1070.46094	1086.0789	1086.5998	1078.4575	1082.9544	**1062.2037**
	Std.Dev.	2.8292E+01	3.0810E+01	2.6844E+01	3.0946E+01	3.2752E+01	20.5547601	3.1393E+01	2.7940E+01	3.0719E+01	3.1363E+01	**1.9629E+01**
f_{10}	Avg.Obj	3801.5258	3351.6007	3389.5945	3295.3197	3434.3011	3224.5379	3359.5377	3224.9761	3345.0967	3299.8143	**3089.4919**
	Std.Dev.	4.9797E+02	4.4743E+02	4.8825E+02	4.7756E+02	4.1815E+02	4.3174E+02	4.2120E+02	4.2713E+02	4.1877E+02	4.7800E+02	**2.4836E+02**
f_{11}	Avg.Obj	5070.6695	4527.6240	4584.0447	4561.1760	4553.2822	4337.1775	4511.6006	4672.1382	4530.1474	4535.4291	**4331.2635**
	Std.Dev.	5.8713E+02	4.8492E+02	5.7978E+02	5.0593E+02	4.6154E+02	4.9193E+02	6.4470E+02	5.1047E+02	5.7817E+02	6.3195E+02	**2.6978E+02**
f_{12}	Avg.Obj	1200.2119	1200.1349	1200.1363	1200.1364	1200.1441	**1200.1079**	1200.1435	1200.1523	1200.1406	1200.1474	1200.1338
	Std.Dev.	2.1980E-02	4.8660E-02	4.8960E-02	6.1927E-02	5.4431E-02	**2.6305E-02**	5.9890E-02	6.5807E-02	5.5673E-02	5.6505E-02	5.8079E-02
f_{13}	Avg.Obj	1300.4522	1300.4224	1300.4156	1300.4171	1300.4176	**1300.2909**	1300.4073	1300.4085	1300.4230	1300.4214	1300.2973
	Std.Dev.	1.0454E-01	9.8495E-02	7.2704E-02	7.5892E-02	8.9786E-02	**6.4423E-02**	8.7061E-02	8.9102E-02	8.6463E-02	8.7856E-02	5.2780E-02
f_{14}	Avg.Obj	1400.2357	1400.2235	1400.2279	1400.2276	1400.2265	1400.1892	1400.2331	1400.1965	1400.2246	1400.2264	**1400.1855**
	Std.Dev.	4.8548E-02	5.1892E-02	4.5982E-02	4.1723E-02	4.0404E-02	2.4688E-02	3.9394E-02	3.1739E-02	3.9736E-02	4.6593E-02	**3.3341E-02**
f_{15}	Avg.Obj	1531.1561	1519.9596	1520.7198	1520.7703	1522.3377	**1509.5628**	1520.7434	1520.9690	1522.0823	1520.6277	1509.5833
	Std.Dev.	5.8988E+00	4.8345E+00	6.2317E+00	5.4328E+00	5.9087E+00	**1.0521E+00**	5.8830E+00	5.5143E+00	4.9035E+00	4.8627E+00	8.8795E-01

Table 3 Comparisons of the CSBO with ten different CSBO variants (CSBO1 to 10) through 50 independent runs on 30 test functions with 30 dimensions, taken from IEEE the CEC2014 using 300,000 *NFEs*. (Continue)

#	Criteria	SBO	CSBO1	CSBO2	CSBO3	CSBO4	CSBO5	CSBO6	CSBO7	CSBO8	CSBO9	CSBO10
f_{16}	Avg.Obj	1612.1994	1612.0431	1611.9800	1612.0814	1611.9415	1611.4896	1612.0364	1612.0185	1612.0411	1611.8635	**1611.3027**
	Std.Dev.	5.4865E-01	5.9309E-01	6.5080E-01	5.8621E-01	6.6894E-01	6.3778E-01	6.7979E-01	6.2530E-01	6.4723E-01	5.3512E-01	**7.0681E-01**
f_{17}	Avg.Obj	822195.2316	568367.7646	590030.4502	562431.3064	580322.4153	100893.9826	567756.4027	561244.1233	577000.1444	577013.8509	**98968.6504**
	Std.Dev.	2.7776E+05	2.6939E+05	3.3216E+05	3.2542E+05	3.1016E+05	6.7044E+04	3.2760E+05	3.1896E+05	3.0717E+05	2.9373E+05	**7.3983E+04**
f_{18}	Avg.Obj	4706.3278	4291.1398	4084.2605	4418.1364	4221.9342	**3926.4762**	4418.3638	4417.3218	4221.4035	4564.7970	3943.6015
	Std.Dev.	3.3311E+03	3.3801E+03	2.9673E+03	3.4114E+03	3.2616E+03	**2.6700E+03**	3.4125E+03	3.4111E+03	3.2618E+03	3.2021E+03	2.6935E+03
f_{19}	Avg.Obj	1965.7796	1921.3427	1919.9956	1919.7919	1918.8599	1913.8906	1920.0158	1922.5307	1918.8286	1925.0599	**1911.3690**
	Std.Dev.	1.6300E+01	1.9360E+01	1.8359E+01	1.7834E+01	1.5972E+01	1.1429E+01	1.7870E+01	2.3953E+01	1.5891E+01	2.7195E+01	**1.8808E+00**
f_{20}	Avg.Obj	16248.66429	14277.8451	14207.0057	14827.2017	13500.3219	6104.6818	14831.9354	14833.0209	13516.7993	14466.2800	**6088.4871**
	Std.Dev.	7.8963E+03	6.9697E+03	6.4488E+03	7.0439E+03	6.0350E+03	3.0018E+03	6.8849E+03	7.0615E+03	6.2768E+03	7.4474E+03	**3.1874E+03**
f_{21}	Avg.Obj	543639.8104	369803.6101	362713.5061	383841.7475	371039.0086	75432.2970	381847.1752	372842.1177	383037.9050	378127.4855	**72886.1629**
	Std.Dev.	2.6735E+05	2.2066E+05	2.2809E+05	2.4804E+05	2.3117E+05	4.7912E+04	2.6640E+05	2.4205E+05	2.4249E+05	2.3121E+05	**4.5934E+04**
f_{22}	Avg.Obj	3426.4103	3124.5227	3156.6042	3190.7943	3108.4092	**3094.5845**	3194.2945	3122.1624	3143.5954	3115.0538	3102.9174
	Std.Dev.	2.0830E+02	2.6283E+02	2.9065E+02	2.5019E+02	3.3152E+02	**2.5973E+02**	2.2820E+02	2.4963E+02	3.2058E+02	2.5747E+02	2.7208E+02
f_{23}	Avg.Obj	2615.2605	**2615.2447**	**2615.2447**	2615.2449	**2615.2447**	**2615.2447**	2615.2450	2615.2449	**2615.2447**	**2615.2447**	**2615.2447**
	Std.Dev.	1.7565E-03	**4.7607E-04**	**5.9109E-04**	7.5238E-04	**5.2702E-04**	**4.6189E-04**	9.6308E-04	9.8819E-04	**5.1321E-04**	**6.7484E-04**	**4.4453E-04**
f_{24}	Avg.Obj	2650.9373	2638.9357	2639.8083	2641.3553	2641.9886	2632.0145	2636.6127	2636.7461	2640.0860	2637.7172	**2631.9794**
	Std.Dev.	1.7631E+01	1.7006E+01	1.4293E+01	1.7025E+01	1.5397E+01	1.4615E+01	1.2515E+01	1.2235E+01	1.7051E+01	1.1974E+01	**1.4797E+01**
f_{25}	Avg.Obj	2736.1301	2727.4144	2726.7907	2727.3267	2727.1638	**2725.3447**	2728.1289	2727.6877	2727.9906	2727.0872	2725.3759
	Std.Dev.	4.8677E+00	6.3964E+00	6.1640E+00	6.5673E+00	6.4363E+00	**6.4855E+00**	6.4405E+00	6.9534E+00	7.1484E+00	6.2015E+00	5.8665E+00
f_{26}	Avg.Obj	2780.1235	2746.3333	2746.3304	2746.3370	2744.3717	2744.3423	2744.3531	2748.3248	2746.3250	2742.3604	**2740.3507**
	Std.Dev.	4.0301E+01	5.0244E+01	5.0219E+01	5.0211E+01	5.0023E+01	5.0024E+01	5.0006E+01	5.0345E+01	5.0225E+01	4.9723E+01	**4.9367E+01**
f_{27}	Avg.Obj	3820.3126	3626.3675	3643.3680	3642.8716	3641.6163	3580.7270	3622.4255	3630.8999	3643.7984	3640.0523	**3542.3573**
	Std.Dev.	2.2162E+02	4.0708E+02	3.8345E+02	4.0206E+02	3.8722E+02	3.6460E+02	3.8957E+02	3.9212E+02	3.8732E+02	3.9983E+02	**3.6342E+02**
f_{28}	Avg.Obj	7565.6498	6648.7526	6728.3243	6683.4976	6699.6037	6326.5310	6748.4478	6713.0450	6701.8127	6686.1776	**6250.0076**
	Std.Dev.	6.8329E+02	6.7414E+02	6.7526E+02	5.5294E+02	6.2433E+02	3.2681E+02	5.0195E+02	6.3611E+02	6.0382E+02	5.4873E+02	**4.3900E+02**
f_{29}	Avg.Obj	2628931.7834	226253.0709	404713.7310	4272.7752	4235.3147	**3986.3597**	226271.0231	4316.7656	4250.0500	4231.8687	4231.4484
	Std.Dev.	4.5078E+06	1.5699E+06	1.9939E+06	4.6714E+02	4.6579E+02	**2.4049E+02**	1.5699E+06	4.8755E+02	4.8357E+02	4.6671E+02	2.3206E+02
f_{30}	Avg.Obj	6235.7902	5847.9366	5832.5031	5858.0177	5899.4723	**5234.2711**	5860.7970	5828.5558	5929.6560	5794.0718	5340.3091
	Std.Dev.	6.1930E+02	5.9070E+02	7.8411E+02	7.9001E+02	8.1078E+02	**5.8965E+02**	7.5649E+02	8.5105E+02	7.6738E+02	8.0145E+02	5.4975E+02
Avg. Time		47.06	46.50	45.90	46.73	45.45	44.65	46.77	46.12	44.97	44.57	**44.52**

Table 4 Comparisons of the CSBO with ten different CSBO variants (CSBO1 to 10) through 50 independent runs on 30 test functions with 30 dimensions, taken from IEEE the CEC2014 using 500,000 *NFEs*.

#	Criteria	SBO	CSBO1	CSBO2	CSBO3	CSBO4	CSBO5	CSBO6	CSBO7	CSBO8	CSBO9	CSBO10
f_1	Avg.Obj	4468231.8068	3566524.7076	3572070.2958	3580252.6082	3485492.8735	**1311712.8167**	3441468.6327	3645022.8968	3542794.3668	3548704.9319	1341800.6218
	Std.Dev.	1.0064E+06	9.3177E+05	1.0385E+06	1.0270E+06	9.5192E+05	**3.3924E+05**	9.6758E+05	1.0258E+06	1.0258E+06	8.9002E+05	3.2025E+05
f_2	Avg.Obj	7834638.3851	207996.9719	216515.3278	210173.7167	219930.5476	195243.0860	211904.5081	217223.1631	221034.6058	225208.4669	**192055.6357**
	Std.Dev.	7.0536E+05	5.7026E+04	6.1608E+04	4.5640E+04	5.3825E+04	2.3078E+04	4.3302E+04	5.4010E+04	4.7133E+04	4.9089E+04	**2.7752E+04**
f_3	Avg.Obj	7731.6111	4648.8149	4574.0084	5054.6544	5025.1417	1360.2328	4242.7835	4686.0038	4955.3314	4619.8579	**1272.0111**
	Std.Dev.	3.5901E+03	2.8455E+03	2.3448E+03	2.8016E+03	2.7668E+03	6.8397E+02	2.3305E+03	2.5117E+03	2.7103E+03	2.2496E+03	**6.1448E+02**
f_4	Avg.Obj	597.1506	544.6468	556.6786	549.9348	551.3595	**487.3989**	545.7996	536.7114	547.5235	545.6787	490.2048
	Std.Dev.	4.4552E+01	4.3862E+01	5.0614E+01	4.6240E+01	4.3665E+01	**4.6155E+01**	4.5698E+01	4.4097E+01	4.3541E+01	3.9737E+01	4.4169E+01
f_5	Avg.Obj	521.0520	520.9868	521.0062	520.9869	521.0094	**520.8996**	520.9961	520.9824	520.9703	520.9544	520.9071
	Std.Dev.	3.1402E-02	1.3788E-01	1.7996E-01	5.0745E-02	1.6392E-01	**1.4783E-01**	1.4422E-01	1.6028E-01	1.3095E-01	1.8143E-01	1.9725E-01
f_6	Avg.Obj	656.0078	653.6947	652.8843	653.6553	652.9476	**641.6717**	653.3993	654.1231	653.1250	653.2944	641.7580
	Std.Dev.	2.8107E+00	3.3489E+00	3.9502E+00	3.5725E+00	4.2675E+00	**4.6114E+00**	3.7435E+00	3.4581E+00	3.6556E+00	3.8704E+00	4.4044E+00
f_7	Avg.Obj	701.1191	700.5497	700.5628	700.5558	700.5661	**700.5095**	700.5559	700.5504	700.5580	700.5577	700.5103
	Std.Dev.	9.7072E-03	7.9532E-02	9.4539E-02	1.1260E-01	1.0226E-01	**1.3281E-01**	9.3046E-02	9.8790E-02	1.1044E-01	9.4711E-02	8.1794E-02
f_8	Avg.Obj	1099.8578	1087.1715	1082.2656	1085.8130	1083.1418	1081.7064	1087.0080	1087.2987	1084.0895	1083.2526	**1064.2131**
	Std.Dev.	3.4908E+01	2.8784E+01	3.4996E+01	3.2971E+01	3.6873E+01	3.3088E+01	3.6327E+01	3.8572E+01	3.5933E+01	3.1990E+01	**2.3714E+01**
f_9	Avg.Obj	1318.5125	1283.8737	1291.5648	1280.5302	1288.9681	1278.1345	1284.7326	1278.5045	1279.9505	1290.2672	**1276.6274**
	Std.Dev.	3.5148E+01	5.6557E+01	4.6193E+01	5.1029E+01	6.0512E+01	5.1858E+01	5.0314E+01	5.2137E+01	4.5629E+01	4.8973E+01	**5.4355E+01**
f_{10}	Avg.Obj	6209.1124	5388.9733	5562.6941	5579.6667	5628.5034	5461.4489	5689.5142	5471.2140	5589.3747	5461.1946	**5247.1737**
	Std.Dev.	7.3454E+02	6.4005E+02	5.8023E+02	6.3134E+02	5.4631E+02	7.0631E+02	5.5772E+02	7.2492E+02	5.6777E+02	6.0122E+02	**5.0984E+02**
f_{11}	Avg.Obj	8268.0729	7684.8324	7658.7485	7872.9371	7951.1424	7648.1166	7654.6474	7650.9491	7753.2547	7729.5731	**7542.1847**
	Std.Dev.	7.8569E+02	7.7937E+02	8.6152E+02	7.6233E+02	6.3876E+02	8.2830E+02	8.7300E+02	8.2695E+02	7.5978E+02	7.7326E+02	**8.4183E+02**
f_{12}	Avg.Obj	1200.5104	1200.1703	1200.1467	1200.1693	1200.1625	1200.1453	1200.1595	1200.1766	1200.1521	1200.1625	**1200.1425**
	Std.Dev.	5.2558E-02	5.6566E-02	3.6377E-02	5.5906E-02	4.0272E-02	3.3953E-02	4.9758E-02	5.5321E-02	5.4753E-02	5.0034E-02	**4.2965E-02**
f_{13}	Avg.Obj	1300.5702	1300.5270	1300.5166	1300.5204	1300.5155	1300.4213	1300.5150	1300.5104	1300.5210	1300.5213	**1300.4190**
	Std.Dev.	6.1996E-02	8.0362E-02	7.7947E-02	9.4696E-02	7.6679E-02	6.8006E-02	8.2841E-02	7.6509E-02	7.4990E-02	8.2556E-02	**5.3808E-02**
f_{14}	Avg.Obj	1400.3233	1400.3100	1400.3088	1400.3017	1400.3061	**1400.2167**	1400.3007	1400.3014	1400.2999	1400.3070	1400.2207
	Std.Dev.	3.2112E-02	3.4766E-02	3.5193E-02	3.4550E-02	3.3486E-02	**2.5870E-02**	3.0496E-02	3.0682E-02	3.6545E-02	3.7363E-02	2.5043E-02
f_{15}	Avg.Obj	1546.1797	1538.8718	1540.4127	1536.9560	1538.2024	1525.2999	1539.1468	1538.0265	1539.4516	1539.0304	**1525.0345**
	Std.Dev.	6.2357E+00	8.3285E+00	7.5809E+00	7.0513E+00	7.1699E+00	1.1348E+00	7.4981E+00	7.2065E+00	8.0294E+00	7.9007E+00	**1.2177E+00**

Table 4 Comparisons of the CSBO with ten different CSBO variants (CSBO1 to 10) through 50 independent runs on 30 test functions with 30 dimensions, taken from IEEE the CEC2014 using 500,000 *NFEs*. (Continue)

#	Criteria	SBO	CSBO1	CSBO2	CSBO3	CSBO4	CSBO5	CSBO6	CSBO7	CSBO8	CSBO9	CSBO10
f_{16}	Avg.Obj	1621.6403	1621.1993	1621.3845	1621.3210	1621.2901	**1619.8254**	1621.4207	1621.3882	1621.3995	1621.1946	1619.9312
	Std.Dev.	7.1383E-01	7.7318E-01	6.4918E-01	6.1428E-01	7.6126E-01	**7.3523E-01**	6.0711E-01	6.6645E-01	6.8881E-01	7.4991E-01	5.8796E-01
f_{17}	Avg.Obj	1204758.7599	930858.0412	973729.3922	970350.9345	1005058.1999	**116922.1230**	960655.5956	952845.5130	1014125.9405	984460.0284	130321.9319
	Std.Dev.	3.3170E+05	3.7188E+05	3.9557E+05	3.5464E+05	3.6108E+05	**5.1520E+04**	3.4673E+05	3.3937E+05	3.6321E+05	3.4726E+05	4.7015E+04
f_{18}	Avg.Obj	5769.0266	3859.4904	4212.3281	3968.2543	3803.0662	3287.0203	3968.6996	3964.9416	3805.9809	3662.9572	**3262.1020**
	Std.Dev.	1.2376E+03	1.4781E+03	1.4121E+03	1.4697E+03	1.5489E+03	1.3675E+03	1.4663E+03	1.4640E+03	1.5463E+03	1.4658E+03	**1.4043E+03**
f_{19}	Avg.Obj	1968.1094	1956.2054	1956.5324	1957.7725	1955.7178	**1922.0968**	1958.4278	1957.5544	1954.7934	1963.4405	1923.2066
	Std.Dev.	3.1111E+01	3.1237E+01	3.2534E+01	3.2195E+01	3.0535E+01	**3.2719E+00**	3.2062E+01	3.1985E+01	3.0570E+01	3.1417E+01	8.5602E+00
f_{20}	Avg.Obj	9837.7746	6893.7886	6754.2900	6772.8073	6461.7068	**2612.4487**	7012.0433	6857.3274	6713.3070	6914.6229	2653.0545
	Std.Dev.	4.7445E+03	3.8124E+03	4.1005E+03	4.1320E+03	3.8761E+03	**4.2191E+02**	4.4531E+03	4.0422E+03	3.9583E+03	4.1219E+03	5.0937E+02
f_{21}	Avg.Obj	830680.1880	695003.6057	683325.3955	693627.2561	725864.4488	107289.5323	664926.0213	661220.0060	676380.3968	679189.6473	**95856.9225**
	Std.Dev.	3.2473E+05	2.6990E+05	2.6380E+05	2.8770E+05	3.1124E+05	4.6760E+04	2.4799E+05	2.4913E+05	2.3851E+05	2.4553E+05	**4.1897E+04**
f_{22}	Avg.Obj	4208.1099	4063.9731	3891.2790	3907.4388	3959.5429	3731.3831	3950.8289	3970.2595	3941.6170	4093.9408	**3695.2999**
	Std.Dev.	3.2813E+02	3.6835E+02	3.3355E+02	3.6252E+02	4.5082E+02	2.7952E+02	4.1802E+02	3.2647E+02	4.0802E+02	3.7339E+02	**2.5273E+02**
f_{23}	Avg.Obj	2644.0924	2644.0061	2644.0059	2644.0060	2644.0060	**2644.0056**	2644.0058	2644.0059	2644.0061	2644.0058	2644.0058
	Std.Dev.	7.3895E-03	8.8980E-04	8.3452E-04	7.2954E-04	6.0704E-04	**4.2775E-04**	6.3798E-04	7.0625E-04	6.8976E-04	5.1336E-04	5.8747E-04
f_{24}	Avg.Obj	2702.8048	2701.1678	2698.6987	2695.8638	2699.8626	2686.9508	2696.8633	2695.2002	2697.3540	2695.1365	**2682.8046**
	Std.Dev.	2.6180E+01	2.2831E+01	2.6327E+01	1.5413E+01	1.9808E+01	1.7974E+01	1.7585E+01	1.5246E+01	1.6286E+01	1.5743E+01	**1.8151E+01**
f_{25}	Avg.Obj	2763.3809	2759.3736	2759.9331	2762.2377	2761.0254	2752.1710	2760.8928	2760.3395	2761.0379	2762.4270	**2750.0479**
	Std.Dev.	8.7552E+00	1.2748E+01	1.2820E+01	1.5465E+01	1.1495E+01	9.7819E+00	1.3373E+01	1.3797E+01	1.2715E+01	1.4710E+01	**1.0510E+01**
f_{26}	Avg.Obj	2798.1967	2784.4872	2778.4889	2787.8363	2782.4912	2777.3282	2783.7303	2785.6958	2789.1061	2789.4850	**2754.2627**
	Std.Dev.	1.4108E+01	3.7033E+01	4.1872E+01	5.5993E+01	3.8853E+01	4.1758E+01	5.8002E+01	5.6678E+01	5.2787E+01	5.2807E+01	**5.0222E+01**
f_{27}	Avg.Obj	4721.1860	4715.9723	4693.7173	4695.2241	4676.3828	**4270.3327**	4692.2391	4671.1345	4670.3474	4684.7051	4284.3435
	Std.Dev.	2.1927E+02	1.4407E+02	1.3787E+02	1.4093E+02	1.3162E+02	**1.4704E+02**	1.4107E+02	1.4562E+02	1.1796E+02	1.4515E+02	2.2539E+02
f_{28}	Avg.Obj	12274.1713	11353.9419	11245.5390	11295.1513	11426.0850	10706.9324	11270.3343	11136.0587	11392.7079	11277.8010	**10469.0137**
	Std.Dev.	1.2712E+03	1.0280E+03	9.8242E+02	1.0285E+03	9.2493E+02	8.9628E+02	9.1204E+02	8.3419E+02	1.1201E+03	9.3164E+02	**7.7053E+02**
f_{29}	Avg.Obj	3770881.0725	1743941.9285	1704986.7471	1633138.8079	1610903.9546	10774.5470	1750795.9958	1743892.2413	1734367.5073	1887934.7471	**10718.7712**
	Std.Dev.	1.2994E+07	8.6069E+06	8.4125E+06	8.0676E+06	7.9528E+06	2.2256E+03	8.6401E+06	8.6069E+06	8.5607E+06	9.3833E+06	**2.0744E+03**
f_{30}	Avg.Obj	17874.7858	17041.0865	17300.8074	17119.2179	16838.9417	**15384.2701**	17057.8306	17200.4780	17266.5241	16954.8482	15500.4421
	Std.Dev.	2.4934E+03	1.3978E+03	1.9548E+03	1.8601E+03	2.0225E+03	**1.0900E+03**	1.6155E+03	1.6067E+03	1.7371E+03	1.2672E+03	1.2102E+03
Avg. Time		163.50	157.89	159.57	160.54	160.82	159.09	163.35	161.73	156.33	158.81	**157.43**

6.2. Qualitative analysis

In this Section, we present the qualitative analysis of the CSBO algorithms based on the 30 benchmark functions employed for an effective evaluation. The sample of the convergence progress of the solution values of the 50 iterations for 30 benchmark functions over each run with the CSBO algorithm are shown in Figure 2a - 2j. The results clearly present the convergence speed rates of the algorithms, in which the picture agent of 30 benchmark functions are represented as f_1, f_9, f_{17}, f_{21}, and f_{28}, respectively.

The Unimodal function, shown in Figure 2a and 2f shows the optimization values for the f_1 function. The CSBO10(Tent) clearly demonstrated has the best performance for this benchmark. The CSBO5-(Logistic) also worked very well, as it ranked second among 11 algorithms.

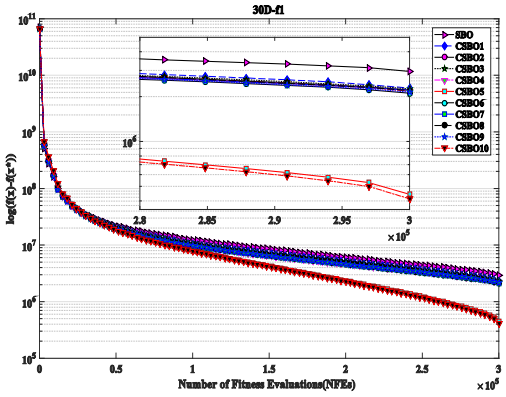
The Simple multimodal function, show in Figure 2b and 2g, shows the optimization values for the f_9 function. The figures reveal that while the differences in performance between the CSBO10(Tent) and CSBO5(Logistic) were slight, the CSBO10(Tent) yielded the better results for this problem, as shown in Tables 3 and 4. Noteably, the convergence to the value of the CSBO5(Logistic) was very close to that of the CSBO10(Tent). Contrastingly, the SBO, CSBO1(Chebyshev), CSBO2(Circle), CSBO3-(Gauss/Mouse), CSBO4(Iterative), CSBO6(Piecewise), CSBO7(Sine), CSBO8(Singer), and CSBO9-(Sinusoidal) did not succeed in this benchmark function.

The hybrid function, shown in Figure 2c, d, h, and i, shows the optimization values for the f_{17} and f_{21} functions; in which the CSBO10(Tent) and CSBO5-(Logistic) yielded the fastest convergence rates of all other methods. Where both were able to more quickly find the optimal solution, the other algo-rithms, upon completion, failed to find the best solution in this benchmark function.

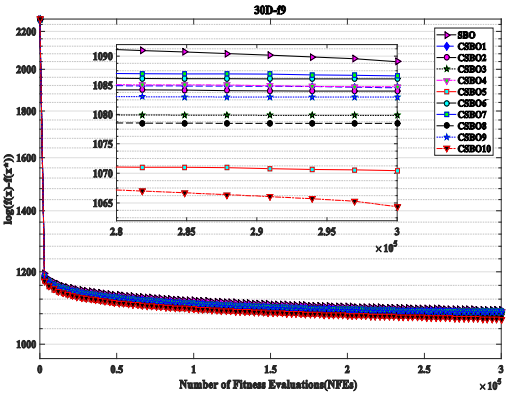
In the composition function, found in Figure 2e and j, all CSBO algorithms demonstrated better search capabilities in finding the optimal solution of each benchmark function. Each algorithm presented sufficient reliability and stability, and was able to search much deeper than the standard SBO algorithm. While they were all capable of converging much faster than the standard algorithm, only the CSBO10(Tent) was able to find the optimal solutions.

Fifty iterations were conducted in order to analyze the convergence curves of the CSBOs on various chaotic maps. Tables 3 and 4 show that on average, the CSBO10(Tent) outperformed the SBO, CSBO1(Chebyshev), CSBO2(Circle), CSBO3-(Gauss/Mouse), CSBO4(Iterative), CSBO5(Logistic), CSBO6(Piecewise), CSBO7(Sine), CSBO8(Singer), and CSBO9(Sinusoidal) algorithms on 21 of the 30 benchmark functions in 30D (f_1 - f_5, f_7 - f_{11}, f_{14}, f_{16}, f_{17}, f_{19} - f_{21}, f_{23}, f_{24}, f_{26} - f_{28}), and 17 out of the 30 benchmark functions in 50D (f_2, f_3, f_8 - f_{13}, f_{15}, f_{18}, f_{21}, f_{22}, f_{24} - f_{26}, f_{28}, f_{29}), when searching for the optimal function minimum. The CSBO5(Logistic) provided the second best map, performing best on 10 out of 30, and 13 out of 30 benchmark functions; based on 30D (f_6, f_{12}, f_{13}, f_{15}, f_{18}, f_{22}, f_{23}, f_{25}, f_{29}, f_{30}) and 50D (f_1, f_4 - f_7, f_{14}, f_{16}, f_{17}, f_{19}, f_{20}, f_{23}, f_{27}, f_{30}), respectively.

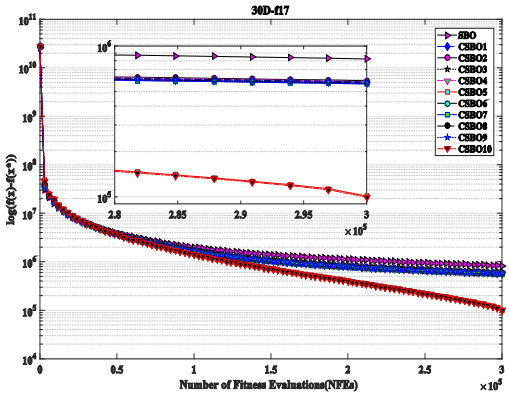
(a) f_1 : Rotated High Conditioned Elliptic in 30D



(b) f_9 : Shifted and Rotated Rastrigin's in 30D

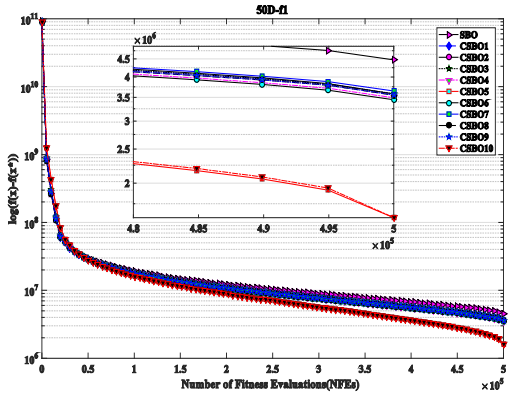


(c) f_{17} : Hybrid 1 (N = 3) in 30D

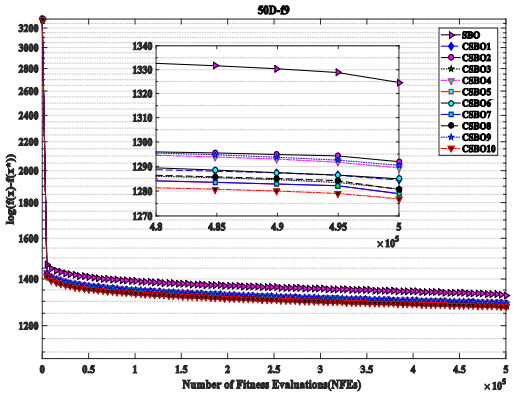


(d) f_{21} : Hybrid 1 (N = 5) in 30D

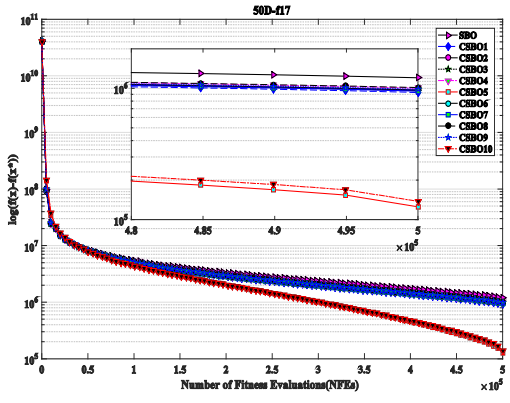
(f) f_1 : Rotated High Conditioned Elliptic in 50D



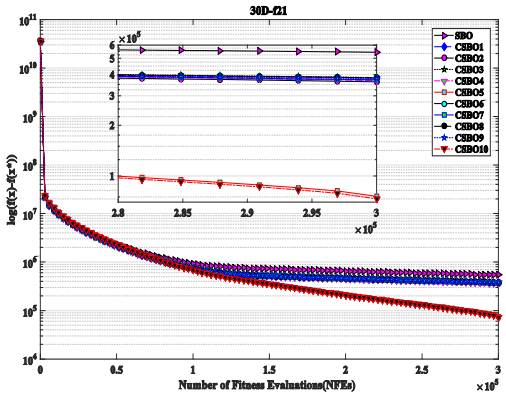
(g) f_9 : Shifted and Rotated Rastrigin's in 50D



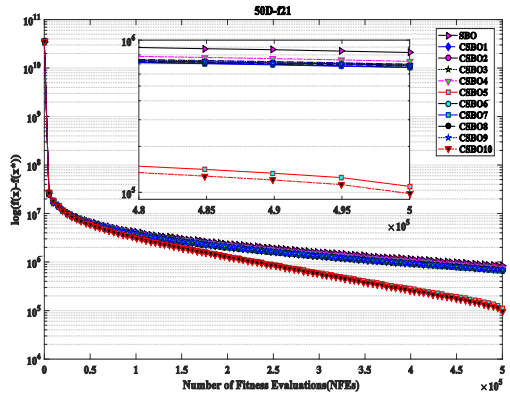
(h) f_{17} : Hybrid 1 (N = 3) in 50D



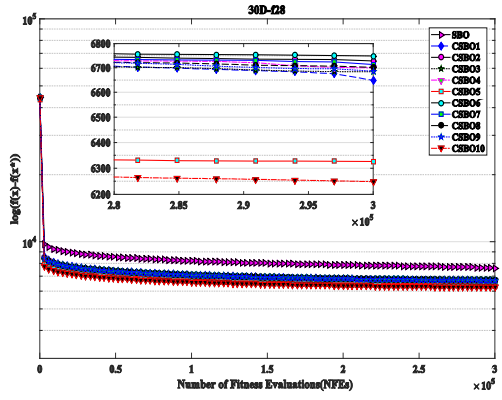
(i) f_{21} : Hybrid 1 (N = 5) in 50D



(e) f_{28} : Composition 7 (N = 3) in 30D



(j) f_{28} : Composition 7 (N = 3) in 50D



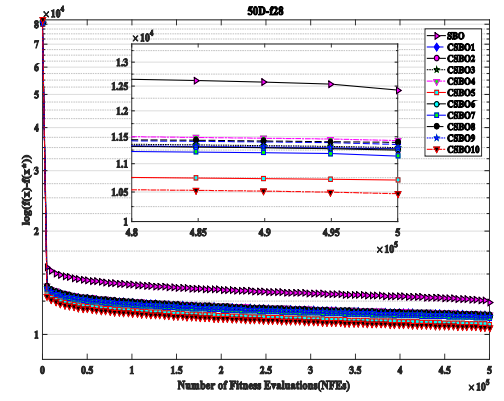


Figure 2 Convergence curves for 30 benchmark functions in 30D and 50D (f_1: Rotated High Conditioned Elliptic Function, f_9: Shifted and Rotated Rastrigin's Function, f_{17}: Hybrid Function 1 (N = 3), f_{21}: Hybrid Function 1 (N = 5), and f_{28}: Composition Function 7, (N = 3).

6.3. Analysis based on the Friedman rank test

The Friedman rank analyses, presented in Tables 5 and 6, reports the performance of the SBO and CSBO algorithms based on the averages of the objective function values of 30 benchmark functions, in both 30 and 50 dimensional cases. Each algo-rithm was ranked according to their performance using an average Friedman rank competition ranking scheme. In competition ranking, algorithms are put in the same rank if their performances are the same. In the simulation results of the average Friedman ranks, the CSBO10(Tent) clearly outper-formed all other algorithms.

Within the 30D problems, the results of the chaotic maps on all benchmark functions displayed the following order: CSBO10(Tent)<CSBO5(Logistic)< CSBO9(Sinusoidal)<CSBO1(Chebyshev)<CSBO2(Circle)<CSBO4(Iterative)<CSBO8(Singer)<CSBO7(Sine)<CSBO6(Piecewise)<CSBO3(Gauss/-Mouse). Within the 50D problems, the results of the chaotic maps on all the benchmark functions displayed the order:CSBO10(Tent)<CSBO5(Logistic)<CSBO7(Sine)<CSBO6(Piecewise)<CSBO2(Circle)<CSBO8(Singer),CSBO9(Sinusoidal<CSBO4-(Iteraive)<CSBO1(Chebyshev)<CSBO3(Gauss-/Mouse). The CSBO10(Tent) exhibited the best (minimum) results, whereas the CSBO3(Gauss/Mouse) yielded the worst (maximum) results. The underlying reason behind the superior performance of CSBO10(Tent) using the Tent chaotic map is that it provides better exploration and local optima avoidance capability.In other words, the Tent map brings different patterns of search behavior to the SBO, resulting in higher exploration capability.

Table 5 Rank table for the mean values of 30 dimensional cases (Friedman rank).

#	SBO	CSBO1	CSBO2	CSBO3	CSBO4	CSBO5	CSBO6	CSBO7	CSBO8	CSBO9	CSBO10
f_1	11.00	10.00	8.00	9.00	5.00	2.00	3.00	7.00	6.00	4.00	**1.00**
f_2	11.00	9.00	5.00	10.00	7.00	2.00	4.00	6.00	3.00	8.00	**1.00**
f_3	11.00	3.00	8.00	10.00	5.00	2.00	7.00	4.00	9.00	6.00	**1.00**
f_4	11.00	9.00	5.00	3.00	4.00	2.00	7.00	10.00	6.00	8.00	**1.00**
f_5	11.00	6.00	8.00	10.00	7.00	2.00	3.00	4.00	9.00	5.00	**1.00**
f_6	11.00	8.00	3.00	4.00	9.00	**1.00**	10.00	5.00	7.00	6.00	2.00
f_7	11.00	7.00	8.00	4.00	5.00	2.00	3.00	9.00	10.00	6.00	**1.00**
f_8	11.00	6.00	2.00	5.00	10.00	3.00	9.00	8.00	4.00	7.00	**1.00**
f_9	11.00	7.00	6.00	4.00	8.00	2.00	9.00	10.00	3.00	5.00	**1.00**
f_{10}	11.00	7.00	9.00	4.00	10.00	2.00	8.00	3.00	6.00	5.00	**1.00**
f_{11}	11.00	4.00	9.00	8.00	7.00	2.00	3.00	10.00	5.00	6.00	**1.00**
f_{12}	11.00	3.00	4.00	5.00	8.00	**1.00**	7.00	10.00	6.00	9.00	2.00
f_{13}	11.00	9.00	5.00	6.00	7.00	**1.00**	3.00	4.00	10.00	8.00	2.00
f_{14}	11.00	4.00	9.00	8.00	7.00	2.00	10.00	3.00	5.00	6.00	**1.00**
f_{15}	11.00	3.00	5.00	7.00	10.00	**1.00**	6.00	8.00	9.00	4.00	2.00
f_{16}	11.00	9.00	5.00	10.00	4.00	2.00	7.00	6.00	8.00	3.00	**1.00**
f_{17}	11.00	6.00	10.00	4.00	9.00	2.00	5.00	3.00	7.00	8.00	**1.00**
f_{18}	11.00	6.00	3.00	8.00	5.00	**1.00**	9.00	7.00	4.00	10.00	2.00
f_{19}	11.00	8.00	6.00	5.00	4.00	2.00	7.00	9.00	3.00	10.00	**1.00**
f_{20}	11.00	6.00	5.00	8.00	3.00	2.00	9.00	10.00	4.00	7.00	**1.00**
f_{21}	11.00	4.00	3.00	10.00	5.00	2.00	8.00	6.00	9.00	7.00	**1.00**
f_{22}	11.00	6.00	8.00	9.00	3.00	**1.00**	10.00	5.00	7.00	4.00	2.00
f_{23}	11.00	**4.00**	**4.00**	8.50	**4.00**	**4.00**	10.00	8.50	**4.00**	**4.00**	**4.00**
f_{24}	11.00	6.00	7.00	9.00	10.00	2.00	3.00	4.00	8.00	5.00	**1.00**
f_{25}	11.00	7.00	3.00	6.00	5.00	**1.00**	10.00	8.00	9.00	4.00	2.00
f_{26}	11.00	8.00	7.00	9.00	5.00	3.00	4.00	10.00	6.00	2.00	**1.00**
f_{27}	11.00	4.00	9.00	8.00	7.00	2.00	3.00	5.00	10.00	6.00	**1.00**
f_{28}	11.00	3.00	9.00	4.00	6.00	2.00	10.00	8.00	7.00	5.00	**1.00**
f_{29}	11.00	8.00	10.00	6.00	4.00	**1.00**	9.00	7.00	5.00	3.00	2.00
f_{30}	11.00	6.00	5.00	7.00	9.00	**1.00**	8.00	4.00	10.00	3.00	2.00
Avg. ranking	11.00	6.20	6.27	6.95	6.40	1.83	6.80	6.72	6.63	5.80	**1.40**
Ranking	11.00	4.00	5.00	10.00	6.00	2.00	9.00	8.00	7.00	3.00	**1.00**

Table 6 Rank table for the mean values of 50 dimensional cases (Friedman rank).

#	SBO	CSBO1	CSBO2	CSBO3	CSBO4	CSBO5	CSBO6	CSBO7	CSBO8	CSBO9	CSBO10
f_1	11.00	7.00	8.00	9.00	4.00	**1.00**	3.00	10.00	5.00	6.00	2.00
f_2	11.00	3.00	6.00	4.00	8.00	2.00	5.00	7.00	9.00	10.00	**1.00**
f_3	11.00	6.00	4.00	10.00	9.00	2.00	3.00	7.00	8.00	5.00	**1.00**
f_4	11.00	4.00	10.00	8.00	9.00	**1.00**	6.00	3.00	7.00	5.00	2.00
f_5	11.00	6.00	9.00	7.00	10.00	**1.00**	8.00	5.00	4.00	3.00	2.00
f_6	11.00	9.00	3.00	8.00	4.00	**1.00**	7.00	10.00	5.00	6.00	2.00
f_7	11.00	3.00	9.00	5.00	10.00	**1.00**	6.00	4.00	8.00	7.00	2.00
f_8	11.00	9.00	3.00	7.00	4.00	2.00	8.00	10.00	6.00	5.00	**1.00**
f_9	11.00	6.00	10.00	5.00	8.00	2.00	7.00	3.00	4.00	9.00	**1.00**
f_{10}	11.00	2.00	6.00	7.00	9.00	4.00	10.00	5.00	8.00	3.00	**1.00**
f_{11}	11.00	6.00	5.00	9.00	10.00	2.00	4.00	3.00	8.00	7.00	**1.00**
f_{12}	11.00	9.00	3.00	8.00	6.50	2.00	5.00	10.00	4.00	6.50	**1.00**
f_{13}	11.00	10.00	6.00	7.00	5.00	2.00	4.00	3.00	8.00	9.00	**1.00**
f_{14}	11.00	10.00	9.00	6.00	7.00	**1.00**	4.00	5.00	3.00	8.00	2.00
f_{15}	11.00	6.00	10.00	3.00	5.00	2.00	8.00	4.00	9.00	7.00	**1.00**
f_{16}	11.00	4.00	7.00	6.00	5.00	**1.00**	10.00	8.00	9.00	3.00	2.00
f_{17}	11.00	3.00	7.00	6.00	9.00	**1.00**	5.00	4.00	10.00	8.00	2.00
f_{18}	11.00	6.00	10.00	8.00	4.00	2.00	9.00	7.00	5.00	3.00	**1.00**
f_{19}	11.00	5.00	6.00	8.00	4.00	**1.00**	9.00	7.00	3.00	10.00	2.00
f_{20}	11.00	8.00	5.00	6.00	3.00	**1.00**	10.00	7.00	4.00	9.00	2.00
f_{21}	11.00	9.00	7.00	8.00	10.00	2.00	4.00	3.00	5.00	6.00	**1.00**
f_{22}	11.00	9.00	3.00	4.00	7.00	2.00	6.00	8.00	5.00	10.00	**1.00**
f_{23}	11.00	9.50	5.50	7.50	7.50	**1.00**	3.00	5.50	9.50	3.00	3.00
f_{24}	11.00	10.00	8.00	5.00	9.00	2.00	6.00	4.00	7.00	3.00	**1.00**
f_{25}	11.00	3.00	4.00	9.00	7.00	2.00	6.00	5.00	8.00	10.00	**1.00**
f_{26}	11.00	6.00	3.00	8.00	4.00	2.00	5.00	7.00	9.00	10.00	**1.00**
f_{27}	11.00	10.00	8.00	9.00	5.00	**1.00**	7.00	4.00	3.00	6.00	2.00
f_{28}	11.00	8.00	4.00	7.00	10.00	2.00	5.00	3.00	9.00	6.00	**1.00**
f_{29}	11.00	8.00	5.00	4.00	3.00	2.00	9.00	7.00	6.00	10.00	**1.00**
f_{30}	11.00	5.00	10.00	7.00	3.00	**1.00**	6.00	8.00	9.00	4.00	2.00
Avg. ranking	11.00	6.65	6.45	6.85	6.63	1.63	6.27	5.88	6.58	6.58	**1.47**
Ranking	11.00	9.00	5.00	10.00	8.00	2.00	4.00	3.00	6.50	6.50	**1.00**

6.4. The non-parametric Wilcoxon's rank sum test

To evaluate the performance of proposed CSBO algorithms, we employed the Wilcoxon's rank sum test, in order to determine the statistical difference of the results achieved by each algorithm. The nonparametric Wilcoxon's rank sum test was conducted for the results obtained by both the SBO and CSBO algorithms, as shown in Tables 7 and 8. The Wilcoxon's rank sum test returns a *p*-Value, which represents the minimal significance level for detecting differences. The level of significance considered was five percent. If the *p*-Value is less than 0.05, it definitively indicates that, in each case, the better result achieved by the best algorithm is statistically significant, and not obtained by chance.

Within the Tables 7 and 8, *N/A* indicates "not applicable", denoting the best objective function value in this current function. In the comparison the SBO and CSBO algorithms, it is generally considered that a *p*-Value of less than 0.05 is sufficient indication of the null hypothesis. The best results are highlighted in bold face, and the

p-Values (greater than 0.050 are underlined).

The comparison summaries of the SBO and among the CSBO in both the 30D and 50D test problems are presented in Tables 3 and 4. From these tables, it is clear that the number of problems in which better average objective fitness values were obtained by the CSBO10(Tent) algorithm were superior to all other algorithms, with the exception of the CSBO5(Logistic). However, the CSBO10-(Tent) results are believed to be biased.

Table 7 Evaluating the results of Table 3 using Wilcoxon's rank sum test.

#	SBO	CSBO1	CSBO2	CSBO3	CSBO4	CSBO5	CSBO6	CSBO7	CSBO8	CSBO9	CSBO10
f_1	7.69E-06	2.23E-09	2.51E-09	3.65E-08	5.06E-09	7.56E-10	6.38E-09	9.63E-10	1.56E-09	3.57E-09	*N/A*
f_2	1.39E-08	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	*N/A*
f_3	1.67E-07	1.56E-09	1.27E-08	1.31E-08	4.78E-09	7.56E-10	9.07E-10	1.38E-09	1.88E-08	4.51E-09	*N/A*
f_4	2.35E-09	8.03E-10	7.56E-10	7.56E-10	8.03E-10	7.56E-10	7.56E-10	1.11E-09	7.56E-10	7.56E-10	*N/A*
f_5	4.55E-06	1.91E-07	3.35E-06	2.93E-08	2.62E-05	1.88E-08	8.31E-07	1.32E-07	6.05E-05	2.09E-06	*N/A*
f_6	4.79E-09	1.20E-08	2.36E-09	1.27E-08	8.21E-08	*N/A*	5.95E-08	3.37E-09	6.99E-08	1.13E-08	1.66E-09
f_7	7.91E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.54E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	*N/A*
f_8	2.96E-09	1.38E-09	7.56E-10	7.56E-10	1.02E-09	7.56E-10	8.03E-10	9.07E-10	7.56E-10	8.03E-10	*N/A*
f_9	6.72E-03	2.49E-01	9.40E-02	2.30E-02	3.37E-01	4.55E-01	2.73E-01	3.18E-01	5.04E-03	8.14E-02	*N/A*
f_{10}	1.10E-08	7.56E-10	9.63E-10	7.56E-10	8.03E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	*N/A*
f_{11}	9.33E-06	5.34E-08	4.64E-06	4.34E-07	8.66E-08	9.00E-09	3.05E-06	1.70E-05	1.12E-06	2.78E-06	*N/A*
f_{12}	1.85E-08	9.63E-10	6.76E-09	4.01E-09	1.38E-09	*N/A*	4.78E-09	1.88E-08	2.82E-09	4.25E-09	2.99E-09
f_{13}	3.20E-06	7.12E-05	8.39E-06	4.43E-06	7.41E-05	*N/A*	5.01E-04	4.03E-04	1.43E-05	2.03E-05	7.78E-08
f_{14}	3.47E-07	2.53E-01	5.08E-01	6.96E-01	4.96E-01	2.51E-01	6.54E-01	3.92E-07	3.47E-01	6.06E-01	N/A
f_{15}	2.27E-04	2.51E-09	1.26E-07	1.59E-08	1.24E-06	*N/A*	5.95E-08	1.07E-07	1.77E-08	4.25E-09	7.56E-10
f_{16}	3.24E-07	2.91E-06	4.79E-08	6.81E-07	2.12E-07	1.56E-09	2.30E-06	2.60E-07	1.17E-06	6.38E-09	*N/A*
f_{17}	1.05E-07	4.07E-08	1.06E-06	1.72E-07	3.37E-07	7.56E-10	2.35E-07	1.26E-07	1.91E-07	1.40E-07	*N/A*
f_{18}	1.48E-03	1.57E-04	4.72E-05	6.67E-04	2.89E-04	*N/A*	6.44E-04	6.44E-04	2.89E-04	8.23E-04	6.39E-06
f_{19}	1.86E-09	2.51E-09	1.86E-09	1.76E-09	1.38E-09	7.55E-10	1.66E-09	3.09E-08	1.38E-09	6.63E-08	*N/A*
f_{20}	3.54E-08	3.57E-09	1.86E-09	1.56E-09	8.03E-10	7.56E-10	1.76E-09	2.79E-09	1.09E-09	4.51E-09	*N/A*
f_{21}	1.30E-09	8.66E-08	7.78E-08	7.91E-07	2.12E-07	7.56E-10	3.35E-06	4.80E-07	6.74E-07	1.72E-07	*N/A*
f_{22}	3.54E-09	4.51E-09	9.53E-09	6.38E-09	1.07E-08	*N/A*	6.38E-09	2.66E-09	5.34E-08	2.99E-09	1.30E-09
f_{23}	1.07E-06	7.56E-10	7.56E-10	7.56E-10	7.56E-10	*N/A*	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10
f_{24}	2.84E-07	8.36E-05	7.67E-06	2.89E-04	1.84E-04	8.66E-08	1.47E-07	1.72E-07	7.72E-05	2.12E-07	*N/A*
f_{25}	2.86E-08	3.17E-09	2.23E-09	5.37E-09	9.00E-09	*N/A*	1.01E-08	1.77E-08	2.48E-08	2.10E-09	1.56E-09
f_{26}	1.50E-04	4.84E-04	4.84E-04	4.84E-04	2.06E-04	2.06E-04	2.06E-04	1.12E-03	4.84E-04	8.71E-05	*N/A*
f_{27}	7.09E-04	1.24E-01	2.37E-01	1.94E-01	1.04E-01	9.79E-01	5.06E-02	1.24E-01	1.46E-01	1.81E-01	*N/A*
f_{28}	1.59E-06	9.07E-10	9.07E-10	7.56E-10	7.56E-10	7.53E-10	7.56E-10	8.53E-10	7.56E-10	7.56E-10	*N/A*
f_{29}	7.30E-05	1.04E-01	4.11E-03	2.54E-02	1.12E-01	*N/A*	1.28E-01	6.21E-03	1.04E-01	1.57E-01	5.04E-01
f_{30}	6.38E-04	2.62E-08	2.78E-06	1.23E-06	2.78E-06	*N/A*	5.31E-07	1.90E-06	5.84E-06	5.87E-07	7.56E-10

Table 8 Evaluating the results of Table 4 using Wilcoxon's rank sum test.

#	SBO	CSBO1	CSBO2	CSBO3	CSBO4	CSBO5	CSBO6	CSBO7	CSBO8	CSBO9	CSBO10
f_1	8.07E-08	6.28E-08	6.81E-07	6.17E-07	3.45E-08	*N/A*	5.64E-08	1.59E-06	4.34E-07	6.28E-08	7.56E-10
f_2	3.86E-09	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.49E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	*N/A*
f_3	4.66E-08	1.02E-07	7.16E-09	3.26E-07	1.13E-07	7.56E-10	3.17E-09	3.65E-08	7.38E-08	4.25E-09	*N/A*
f_4	1.81E-05	5.04E-03	2.59E-01	2.19E-02	4.73E-02	*N/A*	2.24E-02	4.03E-04	8.53E-03	8.53E-03	2.10E-01
f_5	3.49E-09	8.03E-09	1.90E-06	7.56E-10	1.01E-06	*N/A*	2.47E-07	3.45E-08	4.01E-09	9.14E-08	9.02E-10
f_6	8.13E-06	1.01E-08	7.16E-09	1.01E-08	2.82E-09	*N/A*	1.13E-08	2.61E-08	5.37E-09	1.50E-08	7.56E-10
f_7	5.95E-05	7.56E-10	7.56E-10	7.56E-10	7.56E-10	*N/A*	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.54E-10
f_8	1.11E-09	1.56E-09	4.01E-09	1.56E-09	1.07E-08	1.02E-09	3.78E-09	1.34E-08	4.25E-09	2.10E-09	*N/A*
f_9	1.63E-07	1.56E-05	3.38E-05	1.90E-06	2.49E-04	1.23E-06	5.33E-06	1.36E-06	5.31E-07	3.38E-05	*N/A*
f_{10}	2.13E-07	8.03E-10	8.03E-10	8.03E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	8.53E-10	8.03E-10	*N/A*
f_{11}	4.26E-09	7.12E-05	8.52E-04	1.23E-06	1.41E-08	3.89E-04	3.00E-04	4.34E-04	1.20E-05	2.98E-05	*N/A*
f_{12}	7.91E-06	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.55E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	*N/A*
f_{13}	8.17E-09	6.72E-03	1.28E-01	1.06E-01	1.21E-01	9.14E-01	2.01E-01	2.82E-01	1.44E-02	5.53E-02	*N/A*
f_{14}	2.54E-04	6.15E-01	8.51E-01	1.84E-01	5.85E-01	*N/A*	1.44E-01	2.26E-01	1.44E-01	6.26E-01	7.56E-01
f_{15}	1.20E-08	1.23E-06	5.58E-06	2.51E-09	5.64E-08	7.56E-10	3.62E-07	5.95E-08	1.81E-06	1.64E-06	*N/A*
f_{16}	4.65E-08	1.12E-06	1.94E-05	3.92E-07	2.98E-05	*N/A*	3.14E-05	9.18E-06	1.11E-04	1.32E-07	7.56E-10
f_{17}	1.18E-09	1.77E-04	4.17E-05	7.67E-06	4.64E-06	*N/A*	1.00E-05	1.10E-05	2.53E-06	2.65E-06	7.56E-10
f_{18}	4.95E-04	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.54E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10	*N/A*
f_{19}	1.02E-07	1.76E-09	3.17E-09	3.57E-09	2.23E-09	*N/A*	4.52E-09	4.51E-09	1.09E-09	1.20E-08	7.56E-10
f_{20}	9.26E-07	2.12E-05	1.43E-05	1.84E-04	1.25E-05	*N/A*	4.71E-04	7.72E-05	5.57E-05	1.25E-05	7.56E-10
f_{21}	5.21E-07	3.37E-09	1.66E-09	4.25E-09	1.39E-08	7.56E-10	1.38E-09	1.23E-09	1.30E-09	1.30E-09	*N/A*
f_{22}	3.39E-04	8.58E-01	3.39E-03	1.07E-02	2.42E-01	1.30E-01	8.14E-02	7.33E-02	1.04E-01	3.22E-01	*N/A*
f_{23}	1.20E-04	7.56E-10	7.56E-10	7.56E-10	7.54E-10	N/A	7.56E-10	7.56E-10	7.56E-10	7.56E-10	7.56E-10
f_{24}	4.65E-07	4.54E-01	2.24E-02	1.26E-02	2.61E-02	5.84E-01	2.03E-02	1.05E-03	5.41E-02	1.79E-03	*N/A*
f_{25}	1.68E-08	1.68E-08	4.07E-08	1.81E-06	1.77E-08	8.53E-10	7.78E-08	9.14E-08	1.07E-07	2.09E-06	*N/A*
f_{26}	4.23E-04	1.04E-02	1.69E-01	2.48E-02	3.10E-02	6.83E-01	1.38E-01	6.31E-02	2.48E-02	8.29E-03	*N/A*
f_{27}	1.51E-09	8.03E-10	7.56E-10	7.56E-10	7.56E-10	*N/A*	7.56E-10	7.56E-10	7.56E-10	7.56E-10	2.26E-01
f_{28}	3.49E-04	1.86E-09	1.23E-09	2.51E-09	1.09E-09	7.54E-10	8.53E-10	9.63E-10	7.16E-09	1.38E-09	*N/A*
f_{29}	3.86E-04	3.61E-04	2.11E-03	1.11E-04	2.89E-04	7.56E-10	3.61E-04	4.92E-05	1.35E-04	2.28E-04	*N/A*
f_{30}	5.76E-08	7.56E-10	5.41E-09	1.56E-09	2.99E-09	*N/A*	7.56E-10	7.56E-10	8.53E-10	7.56E-10	7.56E-10

7. Conclusions and future scope

In the present paper, we present a novel, and improved meta-heuristic SBO using chaotic maps to solve complex optimization problems. Ten different chaotic maps were investigated to tune the main parameter, the greatest step size (), of the standard SBO; in which a wide variety of chaotic maps were utilized. In the comparison of several different chaotic SBO algorithms, those utilizing the Tent map were selected as the greatest step size (); and through the comparisons of the various chaotic SBO variants, the best CSBO10(Tent) was formed. The simulations demonstrated that the usage of deterministic chaotic signals rather than linearly decreasing values represents an important modification of the SBO method.The experimental results indicated that the tuned SBO significantly enhances the reliability of the global optimality and the quality of the solutions of the newly formed algorithm, due to the application of the deterministic chaotic signals in place of constant values. In order to evaluate an algorithm with its original (SBO) and improved (CSBO) variants, other mathematical benchmark examples were employed. The statistical results and success rates of the CSBO10(Tent) suggest that the tuned algorithms clearly improve the reliability of the global optimality, and further enhance the quality of the results.

The CSBO proved to be simple and easy to implement within all of our applied (and similar type) applications. In future works, we intend to investigate the further capabilities of the CSBO algorithm in solving real-world engineering problems, as well as discrete optimization problems.

Appendix

The Satin Bowerbird Optimizer (SBO) codes used within this paper were retrieved from the resource: https://www.mathworks.com/matlabcentral/fileexchange/62009-satin-bowerbird-optimizer--sbo-2017-

References

1. Yang D, Liu Z, Zhou J. Chaos optimization algorithms based on chaotic maps with different probability distribution and search speed for global optimization. Commun Nonlinear Sci Numer Simulat 2014;19:1229–46.
2. Lorenz EN. Deterministic Nonperiodic Flow. Journal of the Atmospheric Sci-ences 1963;20:130–41.
3. Yuan X, Zhao J, Yang Y, Wang Y. Hybrid parallel chaos optimization algorithm with harmony search algorithm. Applied Soft Computing 2014;17:12–22.
4. Yuan X, Zhang T, Xiang Y, Dai X. Parallel chaos optimization algorithm with migration and merging operation. Applied Soft Com-puting 2015;35:591–604.
5. Huang L, Ding S, Yu S, Wang J, Lu K. Chaos-enhanced Cuckoo search optimiza-tion algorithms for global optimiza-tion. Applied Mathematical Modelling 2016; 40:3860–75.
6. Alatas B. Chaotic bee colony algorithms for global numerical optimization. Expert Syste-ms with Applications 2010;37:5682–7.
7. Baykasoglu A. Design optimization with chaos embedded great deluge algorithm. Applied Soft Computing 2012;12:1055–67.
8. Talatahari S, Farahmand Azar B, Sheikho-leslami R, Gandomi AH. Imperialist competi-tive algorithm combined with chaos for global optimization. Commun Nonlinear Sci Numer Simulat 2012;17:1312–9.
9. Pluhacek M, Senkerik R, Davendra D. Chaos particle swarm optimization with Eensemble of chaotic systems. Swarm and Evolutionary Computation 2015;25:29–35.
10. Gandomi AH, Yun GJ, Yang XS, Talatahari S. Chaos-enhanced accelerated particle swarm optimization. Commun Nonlinear Sci Numer Simulat 2013;18:327–40.
11. Gandomi AH, Yang XS, Talatahari S, Alavi AH. Firefly algorithm with chaos. Commun Nonlinear Sci Numer Simulat 2013;18:89–98.

12. Mirjalili S, Gandomi AH. Chaotic gravitati-onal constants for the gravitational search algorithm. Applied Soft Computing 2017; 53:407–19.
13. Kohli M, Arora S. Chaotic grey wolf optimization algorithm for constrained opti-mization problems. Journal of Computational Design and Engineering 2017.
14. Kaur G, Arora S. Chaotic whale optimization algorithm. Journal of Computational Design and Engineering 2018:275–84.
15. Problems O, Hassanien AE, Bhattacharyya S, Hassanien AE. Chaotic Crow Search Algorithm for Fractional Optimization Problems. Applied Soft Computing 2018.
16. Sayed GI, Hassanien AE, Azar AT. Feature selection via a novel chaotic crow search algorithm. Neural Computing and Applica-tions 2017:1–18.
17. Wangchamhan T, Chiewchanwattana S, Sunat K. Efficient algorithms based on the k-means and Chaotic League Championship Algorithm for numeric, categorical, and mixed-type data clustering. Expert Systems with Applications 2017;90:146–67.
18. Sayed GI, Khoriba G, Haggag MH. A novel chaotic salp swarm algorithm for global optimization and feature selection. Applied Intelligence 2018:1–20. doi:10.1007/s10489-018-1158-6.
19. Wang GG, Guo L, Gandomi AH, Hao GS, Wang H. Chaotic Krill Herd algorithm. Information Sciences 2014;274:17–34.
20. Samareh Moosavi SH, Khatibi Bardsiri V. Satin bowerbird optimizer: A new optimiza-tion algorithm to optimize ANFIS for soft-ware development effort estimation. Engi-neering Applications of Artificial Intelligence 2017;60:1–15.
21. Chintam J, Daniel M. Real-Power Resche-duling of Generators for Congestion Manage-ment Using a Novel Satin Bower-bird Optimization Algorithm. Energies 2018; 11:1–16.
22. Przystalka P, Moczulski W. Methodology of neural modelling in fault detection with the use of chaos engineering. Engineering App-lications of Artificial Intelligence 2015;41:25–40.
23. Hatamlou A, Abdullah S, Hatamlou M. Data Clustering Using Big Bang-Big Crunch Algorithm. Innovative Computing Technology 2011;241:383–8.
24. Xu Q, Wang S, Zhang L, Liang Y. A novel chaos danger model immune algorithm. Commun Nonlinear Sci Numer Simulat 2013;18:3046–60.
25. J.J Liang, B-Y. Qu, P.N. Suganthan. “Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization.” Tech-nical Report 201311, Computational Intelli-gence Laboratory, Zhengzhou University, Zhengzhou, China and Technical Report, Nanyang Technological University, Singa-pore, December 2013.
26. Ghambari S, Rahati A. An improved artificial bee colony algorithm and its application to reliability optimization problems. Applied Soft Computing 2018;62:736–67.
27. Li MD, Zhao H, Weng XW, Han T. A novel nature-in-spired algorithm for optimization: Virus colony search. Advances in Enginee-ring Software 2016;92:65–88.
28. Liu T, Jiao L, Ma W, Shang R. Quantum-behaved particle swarm optimization with collaborative attractors for nonlinear num-erical problems. Commun Nonlinear Sci Numer Simulat 2017;44:167–83.
29. Cui L, Li G, Zhu Z, Lin Q, Wong KC, Chen J, et al. Adaptive multiple-elites-guided composite differential evolution algorithm with a shift mechanism. Information Scien-ces 2018;422:122–43.
30. Tighzert L, Fonlupt C, Mendil B, Fonlupt C, Mendil B. A Set of New Compact Firefly Algorithms. Swarm and Evolutionary Com-putation 2018;40:92–115.
31. Sallam KM, Elsayed SM, Sarker RA, Essam DL. Landscape-based adaptive operator selection mechanism for differential evolu-tion. Information Sciences 2017;418–419: 383–404.
32. Frank Wilcoxon. Individual comparisons by ranking methods. Biometrics Bulletin 1945; 1:80–3.