Neural Comput & Applic (2014) 25:1077–1097
DOI 10.1007/s00521-014-1597-x

ORIGINAL ARTICLE

Biogeography-based optimisation with chaos

Shahrzad Saremi · Seyedali Mirjalili · Andrew Lewis

Received: 5 December 2013 / Accepted: 8 April 2014 / Published online: 24 April 2014

Abstract The biogeography-based optimisation (BBO) algorithm is a novel evolutionary algorithm inspired by biogeography. Similarly, to other evolutionary algorithms, entrapment in local optima and slow convergence speed are two probable problems it encounters in solving challenging real problems. Due to the novelty of this algorithm, however, there is little in the literature regarding alleviating these two problems. Chaotic maps are one of the best methods to improve the performance of evolutionary algorithms in terms of both local optima avoidance and convergence speed. In this study, we utilise ten chaotic maps to enhance the performance of the BBO algorithm. The chaotic maps are employed to define selection, emigration, and mutation probabilities. The proposed chaotic BBO algorithms are benchmarked on ten test functions. The results demonstrate that the chaotic maps (especially Gauss/mouse map) are able to significantly boost the performance of BBO. In addition, the results show that the combination of chaotic selection and emigration operators results in the highest performance.

Keywords Biogeography-based optimisation algorithm · BBO · Chaos · Constrained optimisation · Chaotic maps · Optimisation

S. Saremi · S. Mirjalili (✉) · A. Lewis
School of Information and Communication Technology, Griffith University, Nathan, Brisbane, QLD 4111, Australia
e-mail: seyedali.mirjalili@griffithuni.edu.au

S. Saremi
e-mail: shahrzad.saremi@griffithuni.edu.au

A. Lewis
e-mail: a.lewis@griffith.edu.au

1 Introduction

Recently, evolutionary algorithms (EA) have received much attention. These algorithms have been inspired by different natural evolutionary mechanisms. Some of the most popular algorithms in this field are genetic algorithms (GA) [1], evolutionary strategy (ES), differential evolution (DE) [2], evolutionary programming (EP) [3], genetic programming (GP) [4], probability-based incremental learning (PBIL), and Krill herd (KH) algorithm [5–11]. Regardless of their different structures, these algorithms usually create a random population and evolve it over a predefined number of generations. The evolution process is performed by selection, reproduction, mutation, and recombination operators. The EAs are also similar in dividing the search process into two phases: exploration and exploitation.

The exploration phase occurs when an algorithm tries to discover the promising parts of a search space as extensively as possible [12]. In this phase, the population faces abrupt changes. The EAs are mostly equipped with selection and recombination operators in order to explore search spaces. In contrast, the exploitation phase refers to the convergence towards the most promising solution(s) obtained from the exploration phase as quickly as possible. The population encounters small changes in the exploitation phase. The mutation operators bring exploitation for EAs by randomly manipulating the individuals in the population. In many cases, exploration and exploitation are in conflict and there is no clear boundary between these two phases due to the stochastic nature of EAs [13]. These issues leave EA prone to stagnation in local optima. In other words, EAs may become trapped in local optima without proper balance between exploration and exploitation. There are many studies that focus on improving the performance of EA by boosting exploration and exploitation.

One of the mathematical approaches that recently have been employed to improve both exploration and exploitation is chaos. Chaos theory is related to the study of chaotic dynamical systems that are highly sensitive to initial conditions. One might think that chaotic systems have random behaviour, but providing chaotic behaviour does not necessarily need randomness. Deterministic systems can also show chaotic behaviours [14]. Some of the recent studies that utilise chaos theory in EAs are the following:

In 2001, Wang et al. [15] developed chaotic crossover and mutation operators and proved that chaos theory can improve the performance of GA successfully. In 2012, Yang and Cheng [16] employed chaotic maps to manipulate the mutation probability with the purpose of increasing the exploitation of GA. They proved that this method is able to provide better result compared to the standard GA on three test functions. In 2013, Jothiprakash and Arunkumar [17] created the initial population for GA and DE by chaos theory and applied it to a water resource system problem. They showed that GA and DE with initial chaotic populations outperform those of the standard GA and DE algorithms. The performance of DE was also improved with a chaotic mutation factor by Zhenyu et al. [18]. In 2014, chaotic maps were integrated to the KH algorithm [19, 20]. All these studies show the potential of chaos theory for improving the performance of EAs. This is the motivation of this study: we apply chaos theory to a recently proposed EA called biogeography-based optimisation (BBO) [21].

The BBO algorithm is a novel EA that has been inspired by biogeography [21]. Biogeography studies different ecosystems for finding the relationships between different species in terms of migration and mutation. The inventor of this algorithm, Dan Simon, proved that this algorithm is able to show very competitive results compared to other well-known heuristic and evolutionary algorithms. However, stagnation in local optima and slow convergence speed are two possible problems, similar to other EAs as investigated in the following works:

In 2009, Du et al. [22] combined ES with BBO to improve the local optima avoidance of BBO. They also proposed a new immigration refusal to improve the exploration of the BBO algorithm. In 2010, DE was integrated with BBO, mostly to improve exploitation [23]. Gong et al. [24] proposed a different hybrid of DE and BBO to improve the exploration of the BBO algorithm. In this method, the exploration was done by DE, whereas BBO performed the exploitation. In 2011, Ma and Simon [25] proposed a blended migration operator and indicated that it could improve the exploration of the BBO algorithm. Other improvements, modifications, and hybridisations of BBO can be found in [26–29]. All these studies show that the performance of the BBO algorithm can be further improved.

As mentioned above, one of the methods of improving the performance of EA is by using chaos theory. Currently, there are few works in the literature for improving exploration and exploitation capabilities of BBO using chaos theory. In 2013, we integrated three chaotic maps with BBO and tested it over four test functions [30]. We showed that the chaotic maps can improve the performance of BBO. However, we only integrated chaotic maps with some of the components of the BBO algorithm. This work integrates ten chaotic maps into this algorithm in order to extensively investigate the effectiveness of chaos theory for improving exploration and/or exploitation of the BBO algorithm. Our main motivation for employing chaotic maps for defining emigration and immigration probability is to improve the exploration of BBO. In contrast, chaotic mutation operators are designed to promote exploitation of the BBO algorithm.

The organisation of the paper is as follows: Sect. 2 presents a brief introduction to the BBO algorithm. Section 3 introduces the chaotic maps and proposes the method of combining them with BBO. The experimental results of test functions are provided in Sect. 4. Section 5 concludes the work and suggests some directions for future research.

2 Biogeography-based optimisation algorithm

As its name implies, the BBO algorithm has been inspired by biogeography [21]. The BBO algorithm mimics relationships between different species (habitants) located in different habitats in terms of immigration, emigration, and mutation. In fact, this algorithm simulates the evolution of ecosystems, considering migration and mutation between different geographically separated regions towards a stable situation.

Generally speaking, the concepts of the search process of BBO are identical to those of other evolutionary algorithms, wherein a set of random solutions is first generated. Afterwards, the initial random solutions are evaluated by a fitness function and then evolved over a predefined number of iterations. The BBO algorithm is very similar to GA. Search agents in BBO, called habitats, work similarly to chromosomes in GA. The parameters of habitats, called habitants, are analogous to genes in GA. The associated fitness value for each habitat in BBO is called the Habitat Suitability Index (HSI). Depending on the HSI of habitats, the habitants are able to migrate from one habitat to another. In other words, habitats can be evolved based on their HSI as follows [31]:

- Habitants living in habitats with high HSI are more likely to emigrate to habitats with low HSI
- Habitants located in low-HSI habitats are more prone to allow immigration of new habitants from habitats with high HSI
- Habitats might face random changes in their habitants regardless of their HSI values

These rules assist habitats to improve the HSI of each other and consequently evolving the initial random solutions for a given problem. In the BBO algorithm, each habitat is assigned three rates: immigration (λ_k), emigration (μ_k), and mutation. These rates are calculated based on the number of habitants as follows:

$$\mu_k = \frac{E \times n}{N} \tag{1}$$

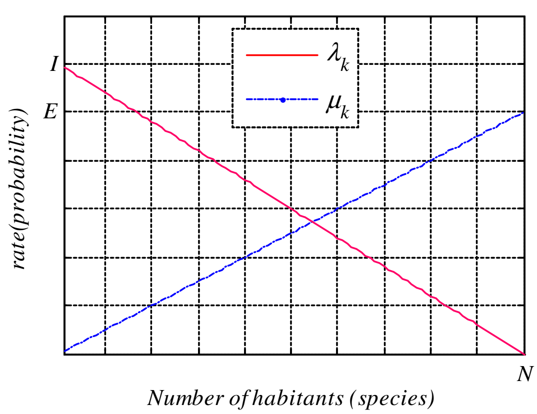

Fig. 1 Emigration (μ_k) and immigration (λ_k) curves

$$\lambda_k = I \times \frac{1-n}{N} \tag{2}$$

where n is the habitant count, N is the maximum number of habitants, E is the maximum emigration rate, and I indicates the maximum immigration rate.

Figure 1 illustrates immigration and emigration rates. This figure shows that the probability of emigration is proportional to the number of habitats. In addition, the immigration probability is inversely proportional to the number of habitants. The mutation rate of BBO is also a function of the number of habitants and defined as follows:

$$m_n = M \times \left(1 - \frac{p_n}{p_{\max}}\right) \tag{3}$$

where M is an initial value for mutation defined by the user, p_n is the mutation probability of the n-th habitat, and $p_{\max} = \arg\max(p_n), \quad n = 1, 2, \ldots, N$.

The pseudocode of the BBO algorithm is illustrated in Fig. 2.

In the next section, the method of integrating the ten chaotic maps is proposed.

3 Chaotic maps for BBO

In this section, we present the chaotic maps used and describe the method of improving the performance of BBO using them.

We chose ten different chaotic maps as shown in Table 1, Fig. 3.

It is worth mentioning here that Fig. 3 shows deterministic systems are also able to provide chaotic

```
Initialize a set of habitats (candidate solutions)
while the termination condition is not satisfied
    Calculate HSI for each habitat
    Update S, λ, and μ of each habitat
    for i=1 to maximum number of habitants do                              Migration
        if rand < λi then                                                  Selection
            for j=1 to maximum number of habitants do
                if rand < μj                                               Emigration
                    Select a random habitant in xi and replace it with xj
                end if
            end for
        end if
    end for
if rand < mutation probability (2.3)                                       Mutation
    Mutate a random number of habitats
end if
Elitism
end while
```

Fig. 2 Pseudocode of the BBO algorithm

Table 1 Chaotic maps

No.	Name	Chaotic map	Range
1	Chebyshev [15]	$x_{i+1} = \cos(i \cos^{-1}(x_i))$	(−1,1)
2	Circle [16]	$x_{i+1} = \mathrm{mod}\left(x_i + b - \left(\frac{a}{2\pi}\right)\sin(2\pi x_k), 1\right)$, $a = 0.5$ and $b = 0.2$	(0,1)
3	Gauss/mouse [17]	$x_{i+1} = \begin{cases} 1 & x_i = 0 \\ \frac{1}{\mathrm{mod}(x_i, 1)} & \text{otherwise} \end{cases}$	(0,1)
4	Iterative [18]	$x_{i+1} = \sin\left(\frac{a\pi}{x_i}\right)$, $a = 0.7$	(−1,1)
5	Logistic [18]	$x_{i+1} = ax_i(1 - x_i)$, $a = 4$	(0,1)
6	Piecewise [19]	$x_{i+1} = \begin{cases} \frac{x_i}{P} & 0 \le x_i < P \\ \frac{x_i - P}{0.5 - P} & P \le x_i < 0.5 \\ \frac{1 - P - x_i}{0.5 - P} & 0.5 \le x_i < 1 - P \\ \frac{1 - x_i}{P} & 1 - P \le x_i < 1 \end{cases}$, $P = 0.4$	(0,1)
7	Sine [20]	$x_{i+1} = \frac{a}{4}\sin(\pi x_i)$, $a = 4$	(0,1)
8	Singer [21]	$x_{i+1} = \mu\left(7.86x_i - 23.31x_i^2 + 28.75x_i^3 - 13.302875x_i^4\right)$, $\mu = 1.07$	(0,1)
9	Sinusoidal [22]	$x_{i+1} = ax_i^2 \sin(\pi x_i)$, $a = 2.3$	(0,1)
10	Tent [23]	$x_{i+1} = \begin{cases} \frac{x_i}{0.7} & x_i < 0.7 \\ \frac{10}{3}(1 - x_i) & x_i \ge 0.7 \end{cases}$	(0,1)

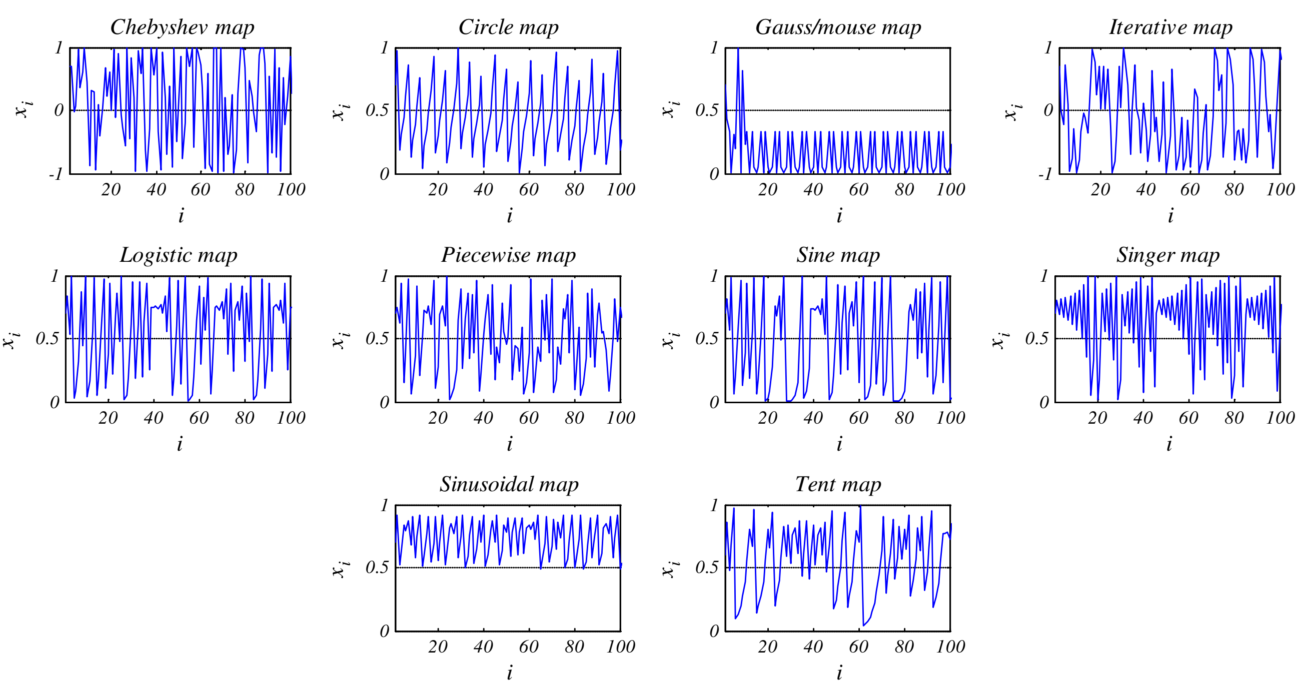

Fig. 3 Visualisation of chaotic maps

behaviours. There is no random component in Table 1, but the chaotic behaviours of the equations are quite evident in Fig. 3. This set of chaotic maps has been chosen with different behaviours, while the initial point is 0.7 for all. The initial point can be chosen any number between 0 and 1 (or −1 and 1 depend 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. In this paper, we use similar initial values to those of [5, 32].

We employ the chaotic maps for manipulating the selection, emigration, and mutation operators of the BBO algorithm. The chaotic selection and emigration operators improve exploration, whereas the chaotic mutation enhances exploitation. A different chaotic map is used for ten variants of the BBO algorithm. The application of chaotic maps is tested singly on each of the main operators and then in combination.

3.1 Chaotic maps for selection

As can be seen in Fig. 2, the selection of a habitat for migration is defined by the probability of λ. We employ the chaotic map to define this probability. The final value from the chaotic map should lie in the interval [0,1], so we normalise those in [−1,1]. In this work, the *rand* value (underlined in Fig. 2) is substituted by values from the chaotic map to provide chaotic behaviours for the selection operator as follows:

$$\begin{array}{l} \textit{if } C(t) < \lambda_i \textit{ then} \\ \quad \textit{Emigrate habitants from } H_i \textit{ to} \\ H_j \textit{ chosen with the probability proportional to } \mu_i \\ \textit{end if} \end{array} \tag{4}$$

where $C(t)$ is the value from the chaotic map in the t-th iteration and H_i shows the i-th habitat.

Note that this method of integrating chaotic maps in the selection phase of BBO is identical to that of [30]. It can be inferred from Eq. (4) that the chaotic maps are responsible for choosing the origin of immigration in our proposed chaotic selection operator.

3.2 Chaotic maps for emigration

As highlighted in Fig. 2, emigration is performed with probability proportional to μ after selecting a habitat. We use the chaotic map to calculate this probability as follows:

$$\begin{array}{l} \textit{if } C(t) < \mu_i \textit{ then} \\ \quad \textit{select a random habitant in } x_i \textit{ and replace it with } x_j \\ \textit{end if} \end{array} \tag{5}$$

where $C(t)$ is the value from the chaotic map in the t-th iteration and x_i shows the i-th habitant.

Equation (5) shows that the chaotic maps are allowed to define the probability of emigration and consequently chaotic emigration behaviours. Note that we again normalise those chaotic maps that lie in the interval [−1,1]

3.3 Chaotic maps for mutation

The probability of mutation is defined directly by the chaotic map as follows:

$$\begin{array}{l} \textit{for } i = 1 \textit{ to number of habitants at k-th habitat} \\ \quad \textit{if } C(t) < \textit{Mutation\_rate}(k) \textit{ then} \\ \quad\quad \textit{Mutate i-th habitants} \\ \quad \textit{end if} \\ \textit{end for} \end{array} \tag{6}$$

where $C(t)$ is the value from the chaotic map in the t-th iteration and *Mutation_rate*(k) shows the mutation rate of k-th habitat, which can be defined by (3) or any other mutation probability generator. Note that the mutation rate is normalised to the range of [0,1].

In the next section, we provide a comparative study using these new chaotic operators and name them chaotic BBO (CBBO). Moreover, combination of these operators (BBO with chaotic selection/emigration and BBO with chaotic selection/emigration/mutation) is also developed and benchmarked.

The following observations may assist in showing how the proposed method operators are theoretically efficient, either individually or together:

- The chaotic selection operator assists CBBO to choose habitats chaotically, which improves exploration
- The chaotic emigration operator allows CBBO to perform emigration with a chaotic pattern that again emphasises exploration
- The chaotic mutation operator helps CBBO to exploit the search space better than BBO since there might be different values for mutation probability
- Different chaotic maps for selection, emigration, and mutation provide different exploration and exploitation patterns for the CBBO algorithm
- Since chaotic maps show chaotic behaviour, a generation of CBBO might emphasise either exploration or exploration
- In case of entrapment in local optima, the chaotic selection and emigration operators combined assist CBBO to exit from them
- In case of finding a promising region(s) of search space, the chaotic mutation operator helps CBBO to chaotically exploit the neighbourhood.

In the following sections, various benchmark functions are employed to demonstrate the effectiveness of the proposed method in action. Note that the source codes of CBBO algorithms can be found in http://www.alimirjalili.com/Projects.html.

4 Experimental results and discussion

To evaluate the performance of the proposed CBBO algorithms, ten standard benchmark functions are employed in this section [3, 33–38]. These benchmark functions are divided into two groups: multimodal and

Table 2 Benchmark functions

Name	Function	Features	Dim	Range
Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$	Unimodal, separable, regular	30	[−5.12, 5.12]
Schwefel	$f(x) = \sum_{i=1}^{n} -x_i sin\left(\sqrt{\lvert x_i \rvert}\right)$	Unimodal, non-separable, regular	30	[−65.536, 64.536]
Rosenbrock	$f(x) = \sum_{i=1}^{n-1}\left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$	Unimodal, non-separable, regular	30	[−2.048, 2.048]
Rastrigin	$f(x) = \sum_{i=1}^{n}\left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	Multimodal, separable, regular	30	[−5.12, 5.12]
Quartic	$f(x) = \sum_{i=1}^{n}\left[ix_i^4\right]$	Unimodal, separable, regular	30	[−1.28, 1.28]
Penalty 1	$f(x) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	Multimodal, non-separable, regular	30	[−50, 50]
Penalty 2	$f(x) = 0.1\left\{sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2\left[1 + sin^2(3\pi x_i + 1)\right] + (x_n - 1)^2\left[1 + sin^2(2\pi x_n)\right]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m x_i > a \\ 0 - a < x_i < a \\ k(-x_i - a)^m x_i < -a \end{cases}$	Multimodal, non-separable, regular	30	[−500, 500]
Griewank	$f(x) = \frac{1}{4{,}000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	Multimodal, non-separable, regular	30	[−600, 600]
Fletcher	$f(x) = \sum_{i=1}^{n}(A_i - B_i)^2$ $A_i = \sum_{j=1}^{n}(a_{ij}\sin\alpha_j + b_{ij}\cos\alpha_j)^2, B_i = \sum_{j=1}^{n}(a_{ij}\sin x_j + b_{ij}\cos x_j)^2$	Multimodal, non-separable, irregular	30	[−π, π]
Ackley	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	Multimodal, non-separable, regular	30	[−50, 50]

unimodal. The former group is suitable for benchmarking exploration, whereas the latter group is use to examine exploitation [35–37]. Table 2 lists these functions and their features, where Dim indicates the dimension of the function, and Range is the boundary of the function's search space. Note that Table 3 provides the initial parameters of the BBO and CBBO algorithms.

Tables 4, 5, 6, 7, and 8 present the experimental results. The results are averaged over ten independent runs. The average (mean) and standard deviation (SD) of the best

Table 3 Initial parameters of BBO and CBBO algorithms

Parameter	Value
Population size	30
Habitat modification probability	1
Immigration probability bounds per gene	[0,1]
Step size for numerical integration of probabilities	1
Maximum immigration (*I*) and maximum emigration (*E*)	1
Mutation probability	0.005

Table 4 Statistical results for chaotic selection operators

Sphere	Mean	SD	*p* values	Schwefel	Mean	SD	*p* values	Rosenbrock	Mean	SD	*p* values
BBO	40.4875	14.46755	0.000183	BBO	5,420.794	834.8779	0.000183	BBO	1,160.35	515.2693	0.000183
CBBO1	23.10065	6.86806	0.000183	CBBO1	4,472.966	522.326	0.000183	CBBO1	680.7491	218.8478	0.000183
CBBO2	44.11793	12.04028	0.000183	CBBO2	5,879.898	763.8601	0.000183	CBBO2	1,173.199	470.9802	0.000183
CBBO3	**7.214329**	**2.087255**	**N/A**	CBBO3	**2,379.449**	**123.8008**	**N/A**	CBBO3	**247.1327**	**50.3269**	**N/A**
CBBO4	37.41302	7.278291	0.000183	CBBO4	5,026.059	640.0222	0.000183	CBBO4	899.1874	378.7307	0.000183
CBBO5	26.25518	7.234038	0.000183	CBBO5	4,412.305	471.1811	0.000183	CBBO5	635.5451	235.635	0.00033
CBBO6	59.65177	17.99541	0.000183	CBBO6	6,390.527	1,003.489	0.000183	CBBO6	1,494.79	368.722	0.000183
CBBO7	22.34138	6.994931	0.000183	CBBO7	4,317.987	486.4445	0.000183	CBBO7	648.4321	296.3546	0.000769
CBBO8	54.73393	13.27617	0.000183	CBBO8	5,937.963	690.0241	0.000183	CBBO8	1,652.609	277.8844	0.000183
CBBO9	122.2511	17.11698	0.000183	CBBO9	8,627.839	529.8165	0.000183	CBBO9	3,891.55	618.0617	0.000183
CBBO10	65.52875	13.88185	0.000183	CBBO10	6,527.814	715.3454	0.000183	CBBO10	1,564.103	590.6724	0.000183
Rastrigin	Mean	SD	*p* values	Quartic	Mean	SD	*p* values	Ackley	Mean	SD	*p* values
BBO	135.1764	41.4407	0.000183	BBO	7.709001	5.201457	0.000183	BBO	16.17653	1.50335	0.000183
CBBO1	70.67161	14.98668	0.000183	CBBO1	3.524945	1.740627	0.000183	CBBO1	15.07637	0.994057	0.000183
CBBO2	136.7921	22.93902	0.000183	CBBO2	8.674079	7.424678	0.000183	CBBO2	15.9084	1.019746	0.000183
CBBO3	**14.78004**	**4.992982**	**N/A**	CBBO3	**0.364736**	**0.202689**	**N/A**	CBBO3	**10.13228**	**0.810754**	**N/A**
CBBO4	92.45411	19.92869	0.000183	CBBO4	6.085191	4.502435	0.000183	CBBO4	15.23403	1.138931	0.000183
CBBO5	86.05691	21.70267	0.000183	CBBO5	4.088008	2.977924	0.000183	CBBO5	14.45722	0.882576	0.000183
CBBO6	164.5355	43.28309	0.000183	CBBO6	15.81948	4.57416	0.000183	CBBO6	17.41293	0.567456	0.000183
CBBO7	69.06214	21.72134	0.000183	CBBO7	3.771994	2.380481	0.000183	CBBO7	14.88165	1.432754	0.000183
CBBO8	180.604	21.83476	0.000183	CBBO8	17.49984	10.26645	0.000183	CBBO8	16.95004	0.943283	0.000183
CBBO9	358.3011	16.4711	0.000183	CBBO9	54.31948	11.31225	0.000183	CBBO9	19.7375	0.20127	0.000183
CBBO10	196.5696	31.83662	0.000183	CBBO10	15.29406	7.366075	0.000183	CBBO10	18.20408	0.516908	0.000183
Fletcher	Mean	SD	*p* values	Griewank	Mean	SD	*p* values	Penalty 1	Mean	SD	*p* values
BBO	6,9312.5	251,581.3	0.000246	BBO	139.8675	49.68852	0.000183	BBO	22,158,883	24,857,860	0.000183
CBBO1	441,597.6	137,055.9	0.001315	CBBO1	99.85251	33.89652	0.000246	CBBO1	2,702,099	2,457,237	0.000769
CBBO2	689,650.2	235,919.2	0.000183	CBBO2	153.3545	55.58816	0.000183	CBBO2	20,818,054	16,124,765	0.000183
CBBO3	**229,011.8**	**69,650.44**	**N/A**	CBBO3	**27.98693**	**8.927177**	**N/A**	CBBO3	**55,428.93**	**70,985.19**	**N/A**
CBBO4	474,282.2	133,291.3	0.001008	CBBO4	90.38315	29.69143	0.000183	CBBO4	14,775,054	7,948,104	0.000183
CBBO5	432,870.7	157,003.4	0.002202	CBBO5	87.10324	16.29776	0.000183	CBBO5	6,008,544	6,479,040	0.000183
CBBO6	924,192	373,056	0.000246	CBBO6	184.9029	48.58532	0.000183	CBBO6	24,137,036	19,521,675	0.000183

Table 4 continued

Fletcher	Mean	SD	p values	Griewank	Mean	SD	p values	Penalty 1	Mean	SD	p values
CBBO7	438,023.8	209,487.4	0.005795	CBBO7	85.46437	36.90809	0.001008	CBBO7	5,622,708	5,719,461	0.000583
CBBO8	704,303.8	187,458.7	0.000183	CBBO8	213.2171	58.89037	0.000183	CBBO8	36,312,131	24,650,653	0.000183
CBBO9	1,782,116	330,261.3	0.000183	CBBO9	394.4245	45.17264	0.000183	CBBO9	2.39E+08	58,226,573	0.000183
CBBO10	780,311.1	206,279.7	0.000183	CBBO10	218.5022	51.85244	0.000183	CBBO10	50,484,341	45,266,446	0.000183

Penalty 2	Mean	SD	p values
BBO	79,650,273	49,005,205	0.000183
CBBO1	37,365,460	29,904,334	0.000183
CBBO2	59,485,096	48,862,529	0.000183
CBBO3	**345,183.7**	**248,726.1**	**N/A**
CBBO4	32,329,998	14,864,342	0.000183
CBBO5	21,280,473	13,171,817	0.000183
CBBO6	1.35E+08	86,117,410	0.000183
CBBO7	9,876,810	10,539,824	0.000183
CBBO8	1.23E+08	76,824,333	0.000183
CBBO9	5.37E+08	1.15E+08	0.000183
CBBO10	1.16E+08	44,201,953	0.000183

Table 5 Statistical results for chaotic emigration operators

Sphere	Mean	SD	p values	Schwefel	Mean	SD	p values	Rosenbrock	Mean	SD	p values
BBO	50.196082	17.886571	0.0001107	BBO	5,234.9902	838.68971	0.0001827	BBO	1,030.369	394.65112	0.0001827
CBBO11	16.485647	3.3495663	0.0001107	CBBO11	3,483.637	513.39965	0.0001827	CBBO11	382.72755	104.3741	0.0001827
CBBO12	75.743784	17.168328	0.0001107	CBBO12	6,583.9246	526.20615	0.0001827	CBBO12	1,508.9449	312.36765	0.0001827
CBBO13	**0.1013361**	**0.3157862**	**N/A**	CBBO13	**486.89234**	**146.39649**	**N/A**	CBBO13	**114.0436**	**39.530465**	**N/A**
CBBO14	46.597883	11.14404	0.0001107	CBBO14	5,796.2697	697.35498	0.0001827	CBBO14	1,176.7902	441.72607	0.0001827
CBBO15	14.290277	4.1343411	0.0001107	CBBO15	3,465.9076	242.10022	0.0001827	CBBO15	395.74362	121.68051	0.0001827
CBBO16	59.078789	15.384816	0.0001107	CBBO16	6,290.7019	693.54658	0.0001827	CBBO16	1,692.9255	738.39404	0.0001827
CBBO17	14.243333	7.0524638	0.0001107	CBBO17	3,285.8978	564.13799	0.0001827	CBBO17	392.78489	60.99782	0.0001827
CBBO18	56.815927	13.527762	0.0001107	CBBO18	6,444.9354	479.71559	0.0001827	CBBO18	1,761.3757	632.81828	0.0001827
CBBO19	158.76119	25.948844	0.0001107	CBBO19	9,783.7476	476.18171	0.0001827	CBBO19	6,198.2429	1,421.2223	0.0001827
CBBO20	52.707267	10.821496	0.0001107	CBBO20	6,350.0215	565.61617	0.0001827	CBBO20	1,653.9094	501.31041	0.0001827
Rastrigin	Mean	SD	p values	Quartic	Mean	SD	p values	Ackley	Mean	SD	p values
BBO	140.26439	39.156	0.0001408	BBO	9.6774181	3.050629	0.0001827	BBO	16.758004	0.8048256	0.0001827
CBBO11	33.375472	7.4156605	0.0001408	CBBO11	1.4296862	1.3396817	0.0001827	CBBO11	12.126539	1.0011187	0.0001827
CBBO12	156.53554	47.935077	0.0001408	CBBO12	16.217893	6.740969	0.0001827	CBBO12	18.418982	0.9112592	0.0001827
CBBO13	**0.4**	**0.5163978**	**N/A**	CBBO13	**0.0029838**	**0.0031468**	**N/A**	CBBO13	**4.632667**	**0.3912018**	**N/A**
CBBO14	115.9659	23.663091	0.0001408	CBBO14	11.104769	4.9685771	0.0001827	CBBO14	16.911048	0.7885035	0.0001827
CBBO15	33.803144	4.1013284	0.0001408	CBBO15	1.6514603	1.1076906	0.0001827	CBBO15	12.614106	1.4969004	0.0001827
CBBO16	142.18916	33.150405	0.0001408	CBBO16	18.862009	6.6133122	0.0001827	CBBO16	18.063982	0.6025753	0.0001827
CBBO17	29.562938	7.826517	0.0001408	CBBO17	1.3993627	0.7309727	0.0001827	CBBO17	12.567542	0.737331	0.0001827
CBBO18	133.19372	33.98165	0.0001408	CBBO18	18.112227	5.6687709	0.0001827	CBBO18	18.038032	0.6201641	0.0001827
CBBO19	201.41873	42.511246	0.0001408	CBBO19	98.342237	22.621424	0.0001827	CBBO19	20.298732	0.3087635	0.0001827
CBBO20	120.30132	33.78048	0.0001408	CBBO20	15.261589	7.8799243	0.0001827	CBBO20	17.592339	0.7567067	0.0001827
Fletcher	Mean	SD	p values	Griewank	Mean	SD	p values	Penalty 1	Mean	SD	p values
BBO	600,415.57	144,902.4	0.0001827	BBO	147.00505	68.22046	0.0001827	BBO	18,033,851	18,407,140	0.0001827
CBBO11	298,838.28	67,675.892	0.0001827	CBBO11	40.37982	10.334234	0.0001827	CBBO11	304,946.77	426,958.7	0.0001827
CBBO12	838,371.25	151,939.73	0.0001827	CBBO12	245.10429	59.433136	0.0001827	CBBO12	80,912,665	37,002,325	0.0001827
CBBO13	**93,607.478**	**35,032.232**	**N/A**	CBBO13	**2.9570968**	**0.5194831**	**N/A**	CBBO13	**1.7187308**	**0.8253504**	**N/A**
CBBO14	700,018.42	222,174.16	0.0001827	CBBO14	160.35023	45.601198	0.0001827	CBBO14	28,280,985	25,018,099	0.0001827
CBBO15	341,246.59	88,665.265	0.0001827	CBBO15	44.086901	17.191089	0.0001827	CBBO15	628,476.71	1,048,770.6	0.0001827
CBBO16	871,870.99	263,032.7	0.0001827	CBBO16	198.23096	49.015071	0.0001827	CBBO16	54,197,696	41,134,473	0.0001827

Table 5 continued

Fletcher	Mean	SD	*p* values	Griewank	Mean	SD	*p* values	Penalty 1	Mean	SD	*p* values
CBBO17	349,643.21	73,865.412	0.0001827	CBBO17	45.443525	11.131416	0.0001827	CBBO17	730,124.34	860,192.32	0.0001827
CBBO18	930,266.11	197,156.15	0.0001827	CBBO18	216.85537	47.592473	0.0001827	CBBO18	44,484,301	17,738,935	0.0001827
CBBO19	2,429,357.7	590,388.12	0.0001827	CBBO19	525.69137	73.447455	0.0001827	CBBO19	470,061,445	115,554,961	0.0001827
CBBO20	929,546.59	200,795.15	0.0001827	CBBO20	179.66274	42.270385	0.0001827	CBBO20	54,097,853	33,943,153	0.0001827

Penalty 2	Mean	SD	*p* values
BBO	58,080,091	23,037,920	0.0001827
CBBO11	4,044,694.5	2,689,388.4	0.0001827
CBBO12	164,564,540	113,326,268	0.0001827
CBBO13	**8.8828837**	**2.1767257**	**N/A**
CBBO14	85,934,893	30,395,774	0.0001827
CBBO15	4,370,520.1	4,510,742.7	0.0001827
CBBO16	178,879,710	99,098,512	0.0001827
CBBO17	3,991,834.9	2,866,476.4	0.0001827
CBBO18	150,436,081	115,420,611	0.0001827
CBBO19	872,725,210	198,475,206	0.0001827
CBBO20	135,386,121	80,371,173	0.0001827

Table 6 Statistical results for chaotic mutation operators

Sphere	Mean	SD	p values	Schwefel	Mean	SD	p values	Rosenbrock	Mean	SD	p values
BBO	50.19608	17.88657	0.427355	BBO	5,614.696	583.4996	0.241322	BBO	1,331.466	691.616	0.307489
CBBO21	55.03699	9.730311	0.025748	CBBO21	**5,220.796**	**646.9606**	**N/A**	CBBO21	1,154.889	411.9002	0.623176
CBBO22	45.2266	15.93448	0.850107	CBBO22	5,553.613	700.136	0.273036	CBBO22	1,115.22	252.0392	0.520523
CBBO23	104.9667	5.19068	0.000183	CBBO23	6,984.822	516.7285	0.000183	CBBO23	2,596.735	493.2488	0.000183
CBBO24	47.60142	17.66667	0.520523	CBBO24	5,875.158	618.665	0.053903	CBBO24	**1,034.089**	**294.1601**	**N/A**
CBBO25	57.38914	18.31544	0.088973	CBBO25	5,792.544	677.7967	0.140465	CBBO25	1,378.675	655.9377	0.161972
CBBO26	52.71786	10.11771	0.121225	CBBO26	5,933.445	608.7155	0.031209	CBBO26	1,590.661	569.6426	0.017257
CBBO27	49.96124	14.51865	0.344704	CBBO27	5,846.311	664.2849	0.053903	CBBO27	1,139.668	300.3623	0.57075
CBBO28	**43.21487**	**12.85429**	**N/A**	CBBO28	5,947.935	601.7959	0.031209	CBBO28	1,230.666	396.4667	0.212294
CBBO29	51.08995	7.755142	0.10411	CBBO29	6,030.586	602.6912	0.021134	CBBO29	1,162.218	383.4772	0.273036
CBBO30	49.677	15.49069	0.344704	CBBO30	6,246.822	729.4931	0.005795	CBBO30	1,553.886	473.7295	0.014019

Rastrigin	Mean	SD	p values	Quartic	Mean	SD	p values	Ackley	Mean	SD	p values
BBO	135.3346	29.70814	0.000246	BBO	9.618467	6.949806	0.57075	BBO	16.93064	1.259177	0.212294
CBBO21	79.50784	18.91382	0.623176	CBBO21	12.0388	7.883459	0.73373	CBBO21	17.10281	0.862088	0.121225
CBBO22	99.67795	21.04073	0.005795	CBBO22	13.46071	7.625377	0.088973	CBBO22	**16.4672**	**0.891572**	**N/A**
CBBO23	91.99247	14.82621	0.031209	CBBO23	32.06718	11.83154	0.000183	CBBO23	18.76668	0.368774	0.000183
CBBO24	79.41879	13.94433	0.384673	CBBO24	11.48982	5.432208	0.472676	CBBO24	16.66166	1.190374	0.73373
CBBO25	77.96833	12.21388	0.57075	CBBO25	11.30964	4.400924	0.241322	CBBO25	16.71692	0.909707	0.520523
CBBO26	143.3328	29.37251	0.000183	CBBO26	11.40406	6.056404	0.520523	CBBO26	17.09329	0.518365	0.140465
CBBO27	**71.26803**	**20.20514**	**N/A**	CBBO27	11.03051	4.781412	0.212294	CBBO27	17.20921	1.00385	0.140465
CBBO28	152.1209	27.8853	0.000183	CBBO28	**8.914222**	**3.067608**	**N/A**	CBBO28	16.68089	0.678343	0.677585
CBBO29	149.8765	35.44308	0.000183	CBBO29	10.97982	4.374899	0.344704	CBBO29	16.96912	0.890759	0.307489
CBBO30	139.7947	20.43994	0.000183	CBBO30	15.8748	5.77396	0.017257	CBBO30	17.05375	0.867548	0.185877

Fletcher	Mean	SD	p values	Griewank	Mean	SD	p values	Penalty 1	Mean	SD	p values
BBO	828,365.4	189,001	0.001315	BBO	142.9973	28.83283	0.57075	BBO	18,595,594	13,784,289	0.73373
CBBO21	689,560	211,055	0.045155	CBBO21	169.7378	48.67349	0.10411	CBBO21	**16,223,763**	**12,454,249**	**N/A**
CBBO22	818,027.3	197,645.3	0.003611	CBBO22	204.1299	73.09895	0.021134	CBBO22	22,196,137	25,646,405	0.96985
CBBO23	1,235,738	248,751.5	0.000183	CBBO23	282.9853	21.98787	0.000183	CBBO23	90,435,214	29,530,896	0.000183
CBBO24	641,463.4	196,436.3	0.121225	CBBO24	**134.6919**	**25.15571**	**N/A**	CBBO24	18,489,499	16,086,231	N/A
CBBO25	802,456	263,524.6	0.021134	CBBO25	177.019	48.90111	0.037635	CBBO25	22,986,907	10,639,030	0.121225
CBBO26	659,477.8	280,430.6	0.427355	CBBO26	152.2939	32.52853	0.273036	CBBO26	30,370,272	20,034,283	0.075662
CBBO27	752,136.1	171,667.5	0.037635	CBBO27	160.4958	44.11886	0.273036	CBBO27	20,178,388	9,199,565	0.241322
CBBO28	633,208.7	243,357.2	0.384673	CBBO28	173.2608	46.9862	0.088973	CBBO28	29,599,824	26,801,567	0.427355
CBBO29	705,897.5	204,904.2	0.075662	CBBO29	167.416	37.24293	0.064022	CBBO29	24,567,458	24,494,912	0.57075
CBBO30	**519,136.7**	**141,286.1**	**N/A**	CBBO30	176.1893	54.53923	0.075662	CBBO30	21,646,813	16,407,386	0.427355

Penalty 2	Mean	SD	p values
BBO	63,937,965	24,464,971	0.307489
CBBO21	59,671,633	35,518,322	0.909722
CBBO22	64,582,529	37,256,642	0.677585
CBBO23	2.49E+08	72,830,612	0.000183
CBBO24	**54,523,474**	**36,957,254**	**N/A**
CBBO25	64,692,549	30,294,479	0.57075
CBBO26	70,087,046	39,756,403	0.384673
CBBO27	81,730,022	71,264,909	0.677585
CBBO28	96,874,370	74,156,405	0.212294
CBBO29	69,334,027	46,266,797	0.520523
CBBO30	64,425,762	39,396,989	0.623176

Table 7 Statistical results for BBO algorithms with chaotic selection and migration operators

Sphere	Mean	SD	*p* values	Schwefel	Mean	SD	*p* values	Rosenbrock	Mean	SD	*p* values
BBO	50.19608	17.88657	8.74E−05	BBO	5,610.797	851.607	0.000183	BBO	930.8098	218.0219	0.000183
CBBO31	1.491478	1.210398	0.00012	CBBO31	1,136.975	212.5692	0.000183	CBBO31	173.1946	69.19807	0.025748
CBBO32	20.66334	5.907142	8.74E−05	CBBO32	3,491.107	562.8696	0.000183	CBBO32	438.3275	137.3741	0.000183
CBBO33	**0.008998**	**0.028454**	**N/A**	CBBO33	**378.057**	**117.3007**	**N/A**	CBBO33	**113.0275**	**40.89889**	**N/A**
CBBO34	14.58082	5.172494	8.74E−05	CBBO34	2,915.743	261.9025	0.000183	CBBO34	392.9358	98.81269	0.000183
CBBO35	9.714966	3.456825	8.74E−05	CBBO35	2,338.083	253.3544	0.000183	CBBO35	289.8989	150.7639	0.000246
CBBO36	32.54473	8.670839	8.74E−05	CBBO36	4,396.742	497.5226	0.000183	CBBO36	731.6902	274.4348	0.000183
CBBO37	10.43516	2.878671	8.74E−05	CBBO37	2,431.085	449.9683	0.000183	CBBO37	321.6705	129.6261	0.000246
CBBO38	16.01169	5.291518	8.74E−05	CBBO38	3,161.139	415.7788	0.000183	CBBO38	426.9347	141.785	0.000183
CBBO39	176.0587	14.30248	8.74E−05	CBBO39	10,209.98	342.8171	0.000183	CBBO39	8,020.31	1,244.161	0.000183
CBBO40	43.8274	7.689304	8.74E−05	CBBO40	5,623.096	595.2363	0.000183	CBBO40	1,544.696	411.812	0.000183
Rastrigin	**Mean**	**SD**	***p* values**	**Quartic**	**Mean**	**SD**	***p* values**	**Ackley**	**Mean**	**SD**	***p* values**
BBO	139.6328	33.07232	0.00011	BBO	9.16428	4.099312	0.000183	BBO	16.20038	1.176895	0.000183
CBBO31	2.555523	1.457233	0.000701	CBBO31	0.040696	0.023953	0.000246	CBBO31	6.641402	0.791907	0.000183
CBBO32	34.01259	7.110147	0.00011	CBBO32	1.923832	0.984691	0.000183	CBBO32	13.91886	0.484858	0.000183
CBBO33	**0.2**	**0.421637**	**N/A**	CBBO33	**0.00243**	**0.003034**	**N/A**	CBBO33	**4.211614**	**0.477186**	**N/A**
CBBO34	22.40201	6.781274	0.00011	CBBO34	0.853981	0.220028	0.000183	CBBO34	12.4184	1.189606	0.000183
CBBO35	11.75655	1.939765	0.000107	CBBO35	0.320599	0.117646	0.000183	CBBO35	11.06417	0.860943	0.000183
CBBO36	53.17919	5.298895	0.00011	CBBO36	4.148092	1.196268	0.000183	CBBO36	15.30352	0.779841	0.000183
CBBO37	14.97826	3.964853	0.000109	CBBO37	0.536663	0.255421	0.000183	CBBO37	10.80577	1.183667	0.000183
CBBO38	29.70879	7.055667	0.00011	CBBO38	1.564381	0.599833	0.000183	CBBO38	12.73956	1.113865	0.000183
CBBO39	426.3227	17.42405	0.00011	CBBO39	118.0102	19.48319	0.000183	CBBO39	20.32849	0.19291	0.000183
CBBO40	104.9162	9.753317	0.00011	CBBO40	14.26552	4.842982	0.000183	CBBO40	17.09972	0.697432	0.000183
Fletcher	**Mean**	**SD**	***p* values**	**Griewank**	**Mean**	**SD**	***p* values**	**Penalty 1**	**Mean**	**SD**	***p* values**
BBO	605,901.2	281,867.2	0.000183	BBO	160.2266	44.44677	0.000183	BBO	12,000,096	6,971,130	0.000183
CBBO31	132,495.1	41,796.51	0.002827	CBBO31	7.011958	2.066846	0.000183	CBBO31	13.00446	7.496075	0.000183
CBBO32	417,728	80,905.37	0.000183	CBBO32	56.25377	13.32446	0.000183	CBBO32	1,059,426	865,486.5	0.000183
CBBO33	**80,273.98**	**22,220.9**	**N/A**	CBBO33	**2.553794**	**0.427672**	**N/A**	CBBO33	**2.162768**	**1.117596**	**N/A**
CBBO34	328,289.3	64,463.74	0.000183	CBBO34	45.67645	10.88821	0.000183	CBBO34	346,179.9	669,258.5	0.000183
CBBO35	242,412.8	83,191.67	0.000183	CBBO35	26.44844	4.283077	0.000183	CBBO35	29,824.71	58,509.42	0.000183
CBBO36	414,229	114,915.3	0.000183	CBBO36	94.602	20.88739	0.000183	CBBO36	3,977,768	4,256,899	0.000183

Table 7 continued

Fletcher	Mean	SD	*p* values	Griewank	Mean	SD	*p* values	Penalty 1	Mean	SD	*p* values
CBBO37	229,141.3	52,696.71	0.000183	CBBO37	28.16329	9.109687	0.000183	CBBO37	33,298.06	35,668.48	0.000183
CBBO38	324,392.1	92,800.71	0.000183	CBBO38	50.46863	6.092503	0.000183	CBBO38	555,385.5	807,059.8	0.000183
CBBO39	2,913,479	532,696.1	0.000183	CBBO39	576.2176	77.88877	0.000183	CBBO39	5.01E+08	1.22E+08	0.000183
CBBO40	645,207.6	160,314.1	0.000183	CBBO40	158.1851	27.70901	0.000183	CBBO40	23,895,706	15,980,709	0.000183

Penalty 2	Mean	SD	*p* values
BBO	56,754,956	32,860,296	0.000183
CBBO31	10,072.41	15,904.39	0.000183
CBBO32	7,208,867	4,798,605	0.000183
CBBO33	**7.710512**	**2.187002**	**N/A**
CBBO34	2,948,364	1,940,692	0.000183
CBBO35	1,052,793	961,556.9	0.000183
CBBO36	21,789,973	13,224,083	0.000183
CBBO37	790,968	792,274.1	0.000183
CBBO38	3,494,414	1,805,367	0.000183
CBBO39	1.01E+09	1.62E+08	0.000183
CBBO40	1.03E+08	44,328,723	0.000183

Table 8 Statistical results for BBO algorithms with chaotic selection, migration, and mutation operators

Sphere	Mean	SD	*p* values	Schwefel	Mean	SD	*p* values	Rosenbrock	Mean	SD	*p* values
BBO	45.35796	19.07265	0.000169	BBO	5,308.627	849.0884	0.000183	BBO	1,089.123	444.2695	0.000183
CBBO41	2.082932	0.969705	0.001056	CBBO41	1,323.317	230.9569	0.00033	CBBO41	158.577	43.04719	0.185877
CBBO42	27.95529	6.613033	0.000169	CBBO42	4,046.654	451.5654	0.000183	CBBO42	518.1816	114.0546	0.000183
CBBO43	**0.600613**	**0.699102**	**N/A**	CBBO43	**756.1622**	**192.0933**	**N/A**	CBBO43	**135.8368**	**33.05768**	**N/A**
CBBO44	19.55993	6.491257	0.000169	CBBO44	3,621.03	630.8057	0.000183	CBBO44	389.0249	101.9297	0.000183
CBBO45	17.29245	3.835345	0.000169	CBBO45	3,413.421	408.0172	0.000183	CBBO45	435.6251	129.0928	0.000183
CBBO46	38.05475	9.172496	0.000169	CBBO46	4,941.202	531.6314	0.000183	CBBO46	853.29	339.6118	0.000183
CBBO47	19.17401	5.527716	0.000169	CBBO47	3,261.25	353.8626	0.000183	CBBO47	436.83	73.3329	0.000183
CBBO48	19.07052	4.522609	0.000169	CBBO48	3,370.889	277.6319	0.000183	CBBO48	445.9152	81.49083	0.000183
CBBO49	171.9544	20.01799	0.000169	CBBO49	10,365.23	286.2265	0.000183	CBBO49	7,353.405	1,173.78	0.000183
CBBO50	80.50798	12.69071	0.000169	CBBO50	7,228.682	493.264	0.000183	CBBO50	1,945.783	448.7321	0.000183
Rastrigin	**Mean**	**SD**	***p* values**	**Quartic**	**Mean**	**SD**	***p* values**	**Ackley**	**Mean**	**SD**	***p* values**
BBO	123.5831	26.49373	0.000129	BBO	11.63246	5.82874	0.000183	BBO	16.78228	0.570958	0.000183
CBBO41	4.200641	2.347819	0.000758	CBBO41	0.075077	0.037307	0.00044	CBBO41	7.414521	0.888698	0.00044
CBBO42	32.7	8.056054	0.000129	CBBO42	2.713506	1.131376	0.000183	CBBO42	14.34876	0.776859	0.000183
CBBO43	**0.3**	**0.483046**	**N/A**	CBBO43	**0.017156**	**0.013788**	**N/A**	CBBO43	**5.298558**	**0.461175**	**N/A**
CBBO44	23.8	7.375636	0.000128	CBBO44	1.506974	0.755279	0.000183	CBBO44	13.28486	1.110219	0.000183
CBBO45	22.73745	4.323335	0.000127	CBBO45	1.280402	0.522666	0.000183	CBBO45	12.55467	1.017179	0.000183
CBBO46	76.51445	10.50744	0.000129	CBBO46	7.161502	2.132844	0.000183	CBBO46	16.31412	0.968846	0.000183
CBBO47	19.5	4.600725	0.000127	CBBO47	1.881594	0.800364	0.000183	CBBO47	13.51434	0.685344	0.000183
CBBO48	33.40305	8.269378	0.000129	CBBO48	1.982967	0.960216	0.000183	CBBO48	13.02129	1.137128	0.000183
CBBO49	431.7354	23.69347	0.000129	CBBO49	119.4053	20.14375	0.000183	CBBO49	20.45506	0.162596	0.000183
CBBO50	126	20.81666	0.000127	CBBO50	26.81091	9.872275	0.000183	CBBO50	18.36942	0.54116	0.000183
Fletcher	**Mean**	**SD**	***p* values**	**Griewank**	**Mean**	**SD**	***p* values**	**Penalty 1**	**Mean**	**SD**	***p* values**
BBO	617,698.9	154,339.5	0.000183	BBO	148.4821	46.40372	0.000183	BBO	10,696,658	5,857,608	0.000183
CBBO41	161,468.5	44,863.39	0.185877	CBBO41	9.914697	2.531724	0.000183	CBBO41	2,481.276	6,539.76	0.000183
CBBO42	449,694.9	132,825	0.000183	CBBO42	78.77828	24.15427	0.000183	CBBO42	3,435,234	2,750,408	0.000183
CBBO43	**130,973.3**	**45,779.15**	**N/A**	CBBO43	**4.145568**	**1.090787**	**N/A**	CBBO43	**3.309994**	**1.013984**	**N/A**
CBBO44	362,869.5	80,678.52	0.000183	CBBO44	63.40352	22.34702	0.000183	CBBO44	845,022.6	1,284,261	0.000183
CBBO45	347,995.4	152,025.5	0.001315	CBBO45	54.30915	12.78053	0.000183	CBBO45	568,818.2	718,314.9	0.000183
CBBO46	490,322.2	78,896.68	0.000183	CBBO46	118.8697	19.08744	0.000183	CBBO46	16,618,758	15,011,553	0.000183

Table 8 continued

Fletcher	Mean	SD	*p* values	Griewank	Mean	SD	*p* values	Penalty 1	Mean	SD	*p* values
CBBO47	395,431.9	124,325.9	0.000183	CBBO47	52.81144	14.60686	0.000183	CBBO47	574,280.8	487,042.9	0.000183
CBBO48	344,480.7	78,610.7	0.000183	CBBO48	60.18287	14.4963	0.000183	CBBO48	778,192.6	855,235.9	0.000183
CBBO49	2,912,990	427,171.7	0.000183	CBBO49	550.3007	49.82247	0.000183	CBBO49	5.12E+08	69,067,958	0.000183
CBBO50	1,133,577	285,649.4	0.000183	CBBO50	262.5888	23.09858	0.000183	CBBO50	93,399,656	34,851,678	0.000183

Penalty 2	Mean	SD	*p* values
BBO	58,161,671	34,532,222	0.000183
CBBO41	13,289.87	20,547.33	0.000246
CBBO42	22,520,153	16,641,699	0.000183
CBBO43	**294.4026**	**596.6577**	**N/A**
CBBO44	4,625,902	3,432,574	0.000183
CBBO45	4,299,153	3,664,086	0.000183
CBBO46	58,219,295	31,140,981	0.000183
CBBO47	7,092,881	5,857,951	0.000183
CBBO48	4,470,527	2,928,758	0.000183
CBBO49	1.05E+09	2.05E+08	0.000183
CBBO50	2.27E+08	1E+08	0.000183

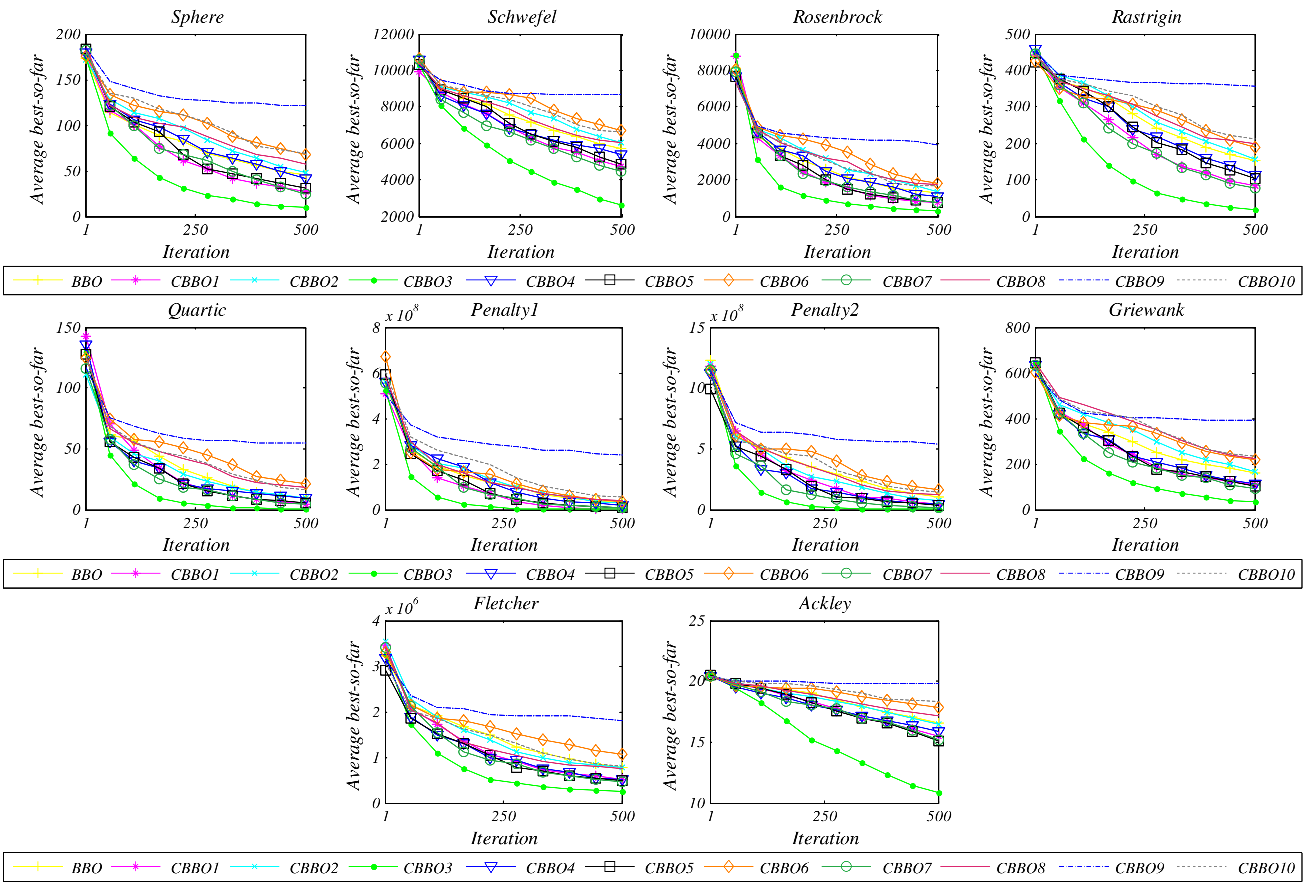

Fig. 4 Convergence curves for BBOs with chaotic selection operators

solution obtained in the last iteration are reflected in the tables. According to Derrac et al. [39], to evaluate the performance of evolutionary algorithms, statistical tests should be conducted. In other words, it is not enough to compare algorithms based on the mean and standard deviation values [40], 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 [41], is carried out at 5 % significance level. 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. It is generally considered that p values <0.05 can be considered as sufficient evidence against the null hypothesis. Note that the best results are highlighted in bold face and the p values >0.05 are underlined.

Table 4 shows the results of the CBBO algorithm with chaotic selection operators. CBBO1 to CBBO10 utilise Chebyshev, Circle, Gauss/mouse, Iterative, Logistic, Piecewise, Sine, Singer, Sinusoidal, and Tent selection operators, respectively. It can be seen in this table that CBBO2, CBBO6, CBBO8, CBBO9, and CBBO10 show worse results compared to the BBO algorithm. This shows that the Circle, Piecewise, Singer, Sinusoidal, and Tent chaotic selection operators are not able to improve the performance of BBO algorithm. In contrast, the CBBO1, CBBO3, CBBO4, CBBO5, and CBBO7 algorithms show much better results compared to BBO on all the test functions. In other words, Chebyshev, Gauss/mouse, Iterative, Logistic, and Sine chaotic maps have successfully improved the performance of the BBO algorithm. Table 4 shows that the Gauss/mouse-based BBO algorithm yields the best results on all of the test functions. The p values reported prove that this superiority is statistically significant. Moreover, Fig. 4 shows the convergence curves of the algorithm. As may be seen in this figure, the Gauss/mouse selection operator has the fastest convergence rate. Considering the results of Table 4 and Fig. 4, it can be stated that the Gauss/mouse maps improve the performance of BBO in terms of both exploration and exploitation.

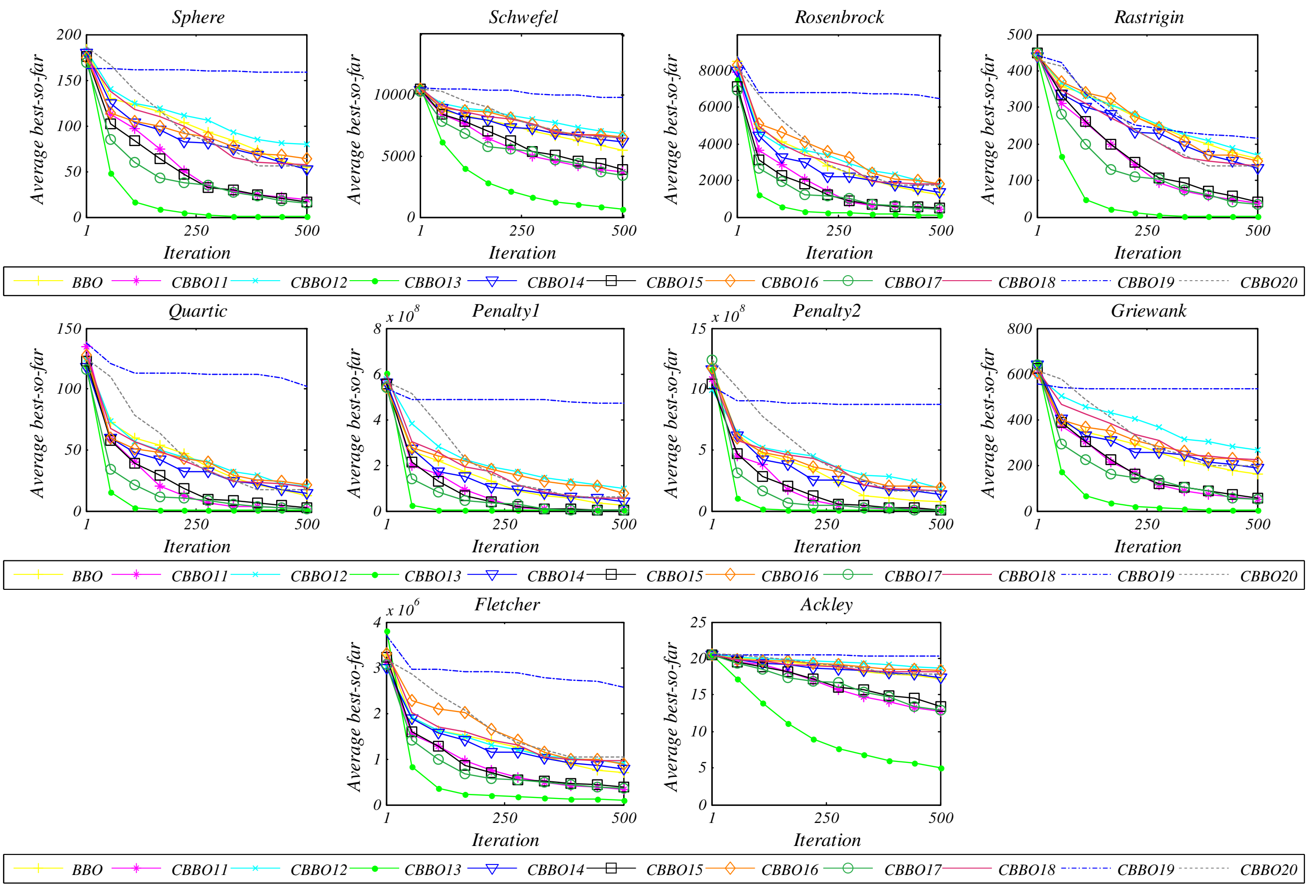

Fig. 5 Convergence curves for BBO algorithms with chaotic migration operators

The statistical results of the CBBO algorithms with chaotic emigration operators are presented in Table 5. As this table shows, CBBO11, CBBO13, CBBO14, VBBO15, and CBBO17 show better results than the BBO algorithm. However, CBBO12, CBBO16, CBBO18, CBBO19, and CBBO20 failed to outperform the BBO algorithm. These results are totally consistent with those of Table 4: again, the Chebyshev, Gauss/mouse, Iterative, Logistic, and Sine chaotic emigration operators successfully improve the performance of the BBO algorithm. Among these operators, the Gauss/mouse chaotic emigration operator once more shows the best statistical results. The *p* values presented prove that this superiority is significant in a statistical way. The convergence curves of CBBO11 to CBBO20 and that of BBO are depicted in Fig. 5. This figure shows that CBBO11, CBBO13, CBBO14, VBBO15, and CBBO17 have the fastest convergence rates. Note that the CBBO13 algorithm shows the overall fastest convergence speed as well. This indicates that the performance of BBO can be boosted by the Gauss/mouse emigration operator in terms of not only exploration but also exploitation.

Table 6 shows the results of the chaotic mutation operators. In contrast to the previous results, there is no absolute superiority for any of the algorithms. For instance, CBBO28 and CBBO21 provide the best results on Sphere and Schwefel functions, whereas CBBO24 and CBBO27 show the best results on Rosenbrock and Rastrigin functions, respectively. There is also no distinguishing convergence behaviour for the algorithms; almost all algorithms are clustered together in their performance, as shown in Fig. 6. However, the results of CBBO22, CBBO24, CBBO25, CBBO27, CBBO28, and CBBO30 are mostly better than the BBO algorithm. This suggests that chaotic mutation operators may also be able to improve the performance of the BBO algorithm, but none of them can provide significant superiority as *p* values of Table 6 confirm (the *p* values underlined are those >0.05).

The results of the CBBO algorithms with both chaotic selection and emigration operators are provided in Table 7 and Fig. 7. It can be observed that the results of all CBBO algorithms are much better than those of Tables 4, 5, and 6. All of the CBBO algorithms provide superior results compared to the BBO algorithm except CBBO39. The convergence curves in Fig. 7 also indicate that the BBO and CBBO39 algorithms provide the worst rates. According to Table 7 and Fig. 7, CBBO33 which utilises the Gauss/mouse selection and emigration operators remarkably enhance the performance of the BBO algorithm. Since

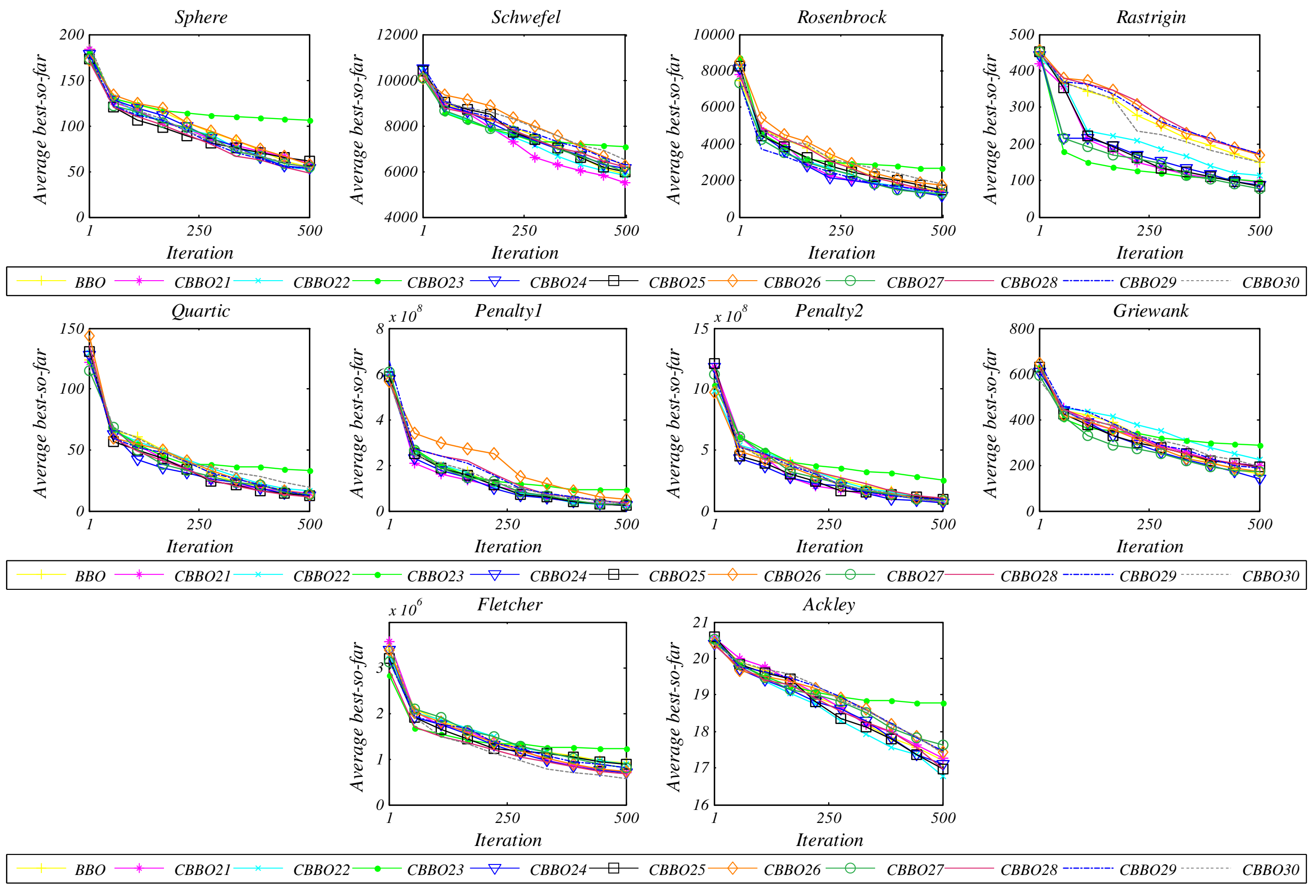

Fig. 6 Convergence curves for BBO algorithms with chaotic mutation operators

CBBO33 provides the best results on all the test functions such as unimodal and multimodal, it can be stated that the combination of the Gauss/mouse selection and emigration operators significantly improves the exploration and exploitation of the BBO algorithm.

Table 8 and Fig. 8 show the experimental results of CBBO41 to CBBO50. As Table 8 suggests, all CBBO algorithms outperform the BBO algorithm except CBBO49 and CBBO50. It is worth mentioning that the results of this table are worse than those of Table 7. This means that the use of chaotic mutation operators degraded the effects of chaotic selection and emigration operators. Therefore, it can be said that the chaotic selection and emigration operators are more effective than the chaotic mutation operators in terms of improved performance. The CBBO43 algorithm shows the best results on all benchmark functions, proving again that the Gauss/mouse selection and emigration operators are able to improve the performance of the BBO algorithm markedly. The convergence behaviours of the algorithms also support this statement, as shown in Fig. 8.

To sum up, the results show that the Chebyshev, Circle, Gauss/mouse, Iterative, Logistic, Piecewise, Sine, Singer, Sinusoidal, and Tent selection, emigration, and mutation operators are able to improve the performance of the BBO algorithm. Generally, the results of the chaotic selection and emigration operators are much better than the chaotic mutation operators. This shows that the exploration of the BBO algorithm is weaker than its exploitation. The results prove that chaotic selection and emigration are able to alleviate this weakness.

The results of the CBBO algorithms with both chaotic selection and emigration operators show that the majority of these operators (except Sinusoidal selection and emigration operators) can improve the performance of the BBO algorithm significantly. This is due to boosting the exploration of the BBO algorithm utilising these operators simultaneously, whereby the BBO algorithm can avoid local optima much better. In other words, the chaotic selection and emigration operators bring different patterns of migration behaviour for habitats, which results in showing higher exploration capability.

Generally speaking, the results of the chaotic maps on all the benchmark functions follow the order of Gauss/mouse < Sine < Chebyshev < Logistic < Iterative < normal BBO < Circle < Singer < Piecewise < Tent <

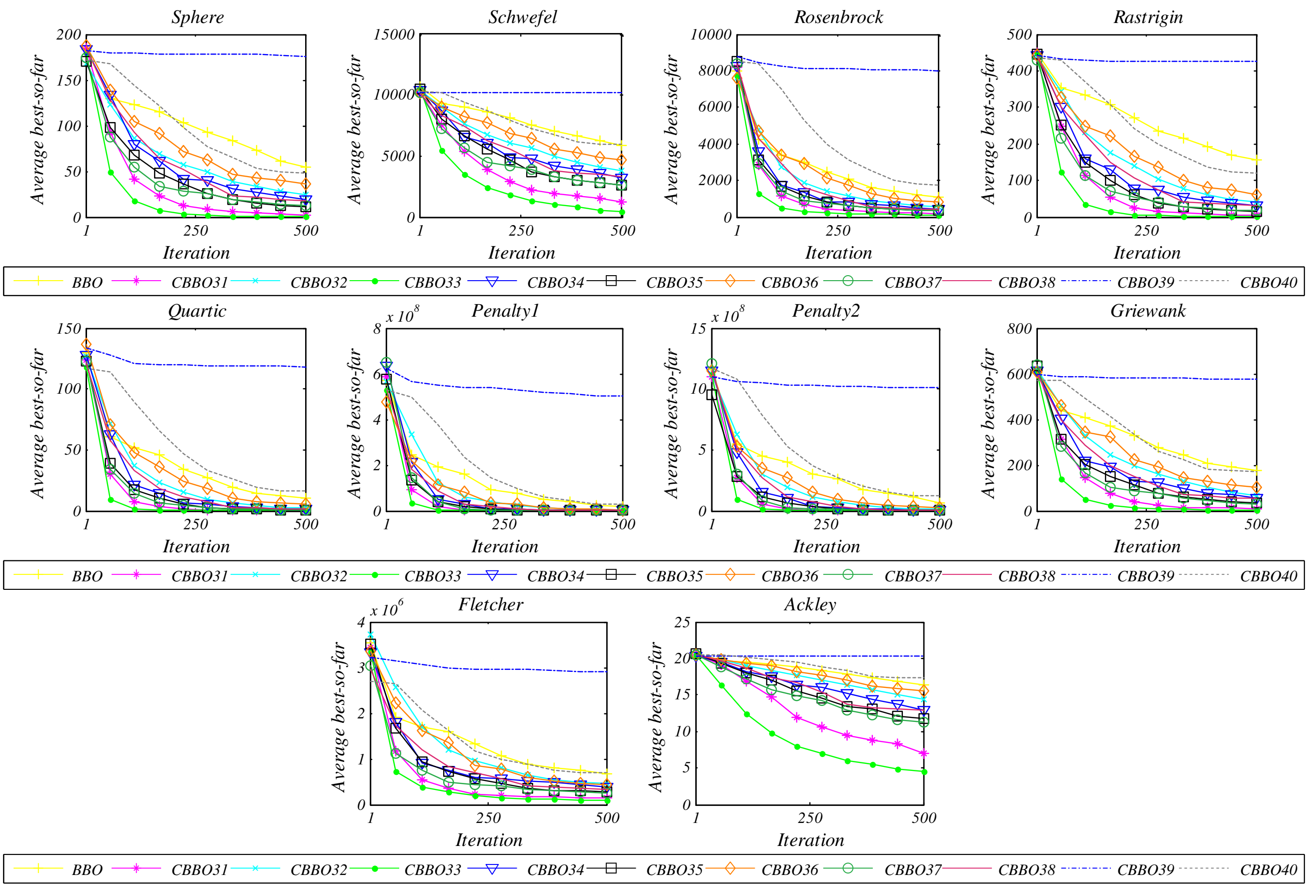

Fig. 7 Convergence curves for BBO algorithms with chaotic selection and emigration operators

Sinusoidal. It can be seen that the Gauss/mouse map shows the best (minimum) results, whereas the Sinusoidal map provides the worst (maximum) results. Matching these results with Fig. 3, it can be seen that the Sinusoidal map usually returns values >0.5. This means that the Sinusoidal map provides less probability of selection, emigration, and mutation compared to others, from consideration of Eqs. (4), (5), and (6). This is the reason for the poor performance of the CBBO9, CBBO19, CBBO29, CBBO39, and CBBO49 variants. In contrast, the Gauss/mouse map has a totally different behaviour, as illustrated in Fig. 3. This chaotic map mostly returns values <0.5. This causes a high probability of emigration over the course of iterations compared to the BBO and other CBBO algorithms. In other words, there is an emphasis on exploration using the Gauss/mouse map that prevents CBBO3, CBBO13, and CBBO33 from stagnating in local optima. The results of unimodal test functions and convergence curves also prove that the superior exploration of the Gauss/mouse map does not have a negative impact on the exploitation.

5 Conclusion

This paper investigated the effectiveness of ten different chaotic maps in improving the performance of the BBO algorithm. In order to compare the chaotic maps in terms of enhancing exploration and exploitation, ten benchmark functions dividing into multimodal and unimodal problems were employed. Generally speaking, the results proved that chaotic maps are able to significantly improve the performance of BBO. Moreover, the results of the comparative study suggest that the Gauss/mouse map is the best map. The reason was the higher selection and emigration probabilities when utilising this map. The results also showed that chaotic mutation operators are not able to improve the performance of the BBO algorithm significantly. Another finding of this paper was that the combination of the chaotic selection and emigration operators is better than other combinations investigated. This was due to the superior exploration of the CBBO with these two operators combined.

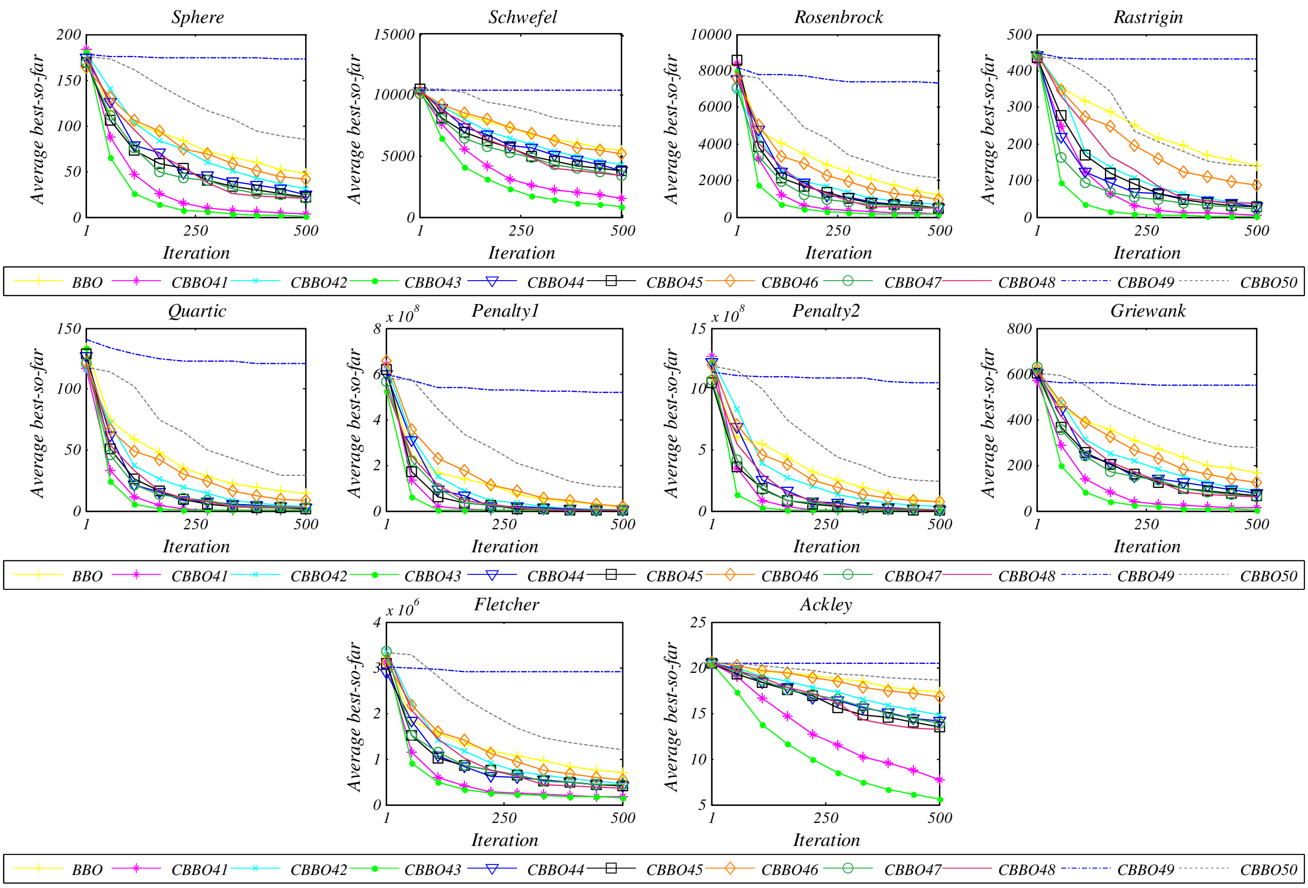

Fig. 8 Convergence curves for BBO algorithms with chaotic selection, emigration, and mutation operators

For future studies, it would be interesting to employ CBBO algorithms for solving real-world engineering problems. In addition, other chaotic maps are also worth applying to BBO.

References

1. John H (1992) Holland, adaptation in natural and artificial systems. MIT Press, Cambridge, MA
2. Price KV, Storn RM, Lampinen JA (1997) Differential evolution. Springer, Berlin
3. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3:82–102
4. Banzhaf W, Koza J, Ryan C, Spector L, Jacob C (2000) Genetic programming. IEEE Intell Syst Appl 15:74–84
5. Gandomi AH, Alavi AH (2012) Krill herd: a new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 17:4831–4845
6. Guo L, Wang G-G, Gandomi AH, Alavi AH, Duan H (2014) A new improved krill herd algorithm for global numerical optimization. Neurocomputing. doi:10.1016/j.neucom.2014.01.023
7. Wang G, Guo L, Wang H, Duan H, Liu L, Li J (2014) Incorporating mutation scheme into krill herd algorithm for global numerical optimization. Neural Comput Appl 24(3-4):853–871. doi:10.1007/s00521-012-1304-8
8. Wang G-G, Gandomi AH, Alavi AH (2013) A chaotic particle-swarm krill herd algorithm for global numerical optimization. Kybernetes 42:9
9. Wang G-G, Gandomi AH, Alavi AH (2013) An effective krill herd algorithm with migration operator in biogeography-based optimization. Appl Math Model. doi:10.1016/j.apm.2013.10.052
10. Wang G-G, Gandomi AH, Alavi AH (2014) Stud krill herd algorithm. Neurocomputing 128:363–370
11. Wang G-G, Gandomi AH, Alavi AH, Hao G-S (2013) Hybrid krill herd algorithm with differential evolution for global numerical optimization. Neural Comput Appl 1–12. doi:10.1007/s00521-013-1485-9
12. Mirjalili S, Mohd Hashim SZ, Moradian Sardroudi H (2012) Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm. Appl Math Comput 218:11125–11137
13. Mirjalili S, Lewis A, Mirjalili SM (2014) Adaptive gbest-guided gravitational search algorithm. Neural Comput Appl 1–19. doi:10.1007/s00521-014-1585-1
14. Kellert SH (1993) In the wake of chaos: unpredictable order in dynamical systems. University of Chicago Press, Chicago
15. Wang N, Liu L, Liu L (2001) Genetic algorithm in chaos. OR Trans 5:1–10
16. Yang L-J, Chen T-L (2002) Application of chaos in genetic algorithms. Commun Theor Phys 38:168–172
17. Jothiprakash V, Arunkumar R (2013) Optimization of hydropower reservoir using evolutionary algorithms coupled with chaos. Water Resour Manag 27(7):1963–1979. doi:10.1007/s11269-013-0265-8

18. Zhenyu G, Bo C, Min Y, Binggang C (2006) Self-adaptive chaos differential evolution. In: Jiao L, Wang L, Gao X, Liu J, Wu F (eds) Advances in natural computation. Lecture notes in computer science, vol 4221. Springer, Berlin, Heidelberg, pp 972–975
19. Saremi S, Mirjalili SM, Mirjalili S (2014) Chaotic Krill herd optimization algorithm. Procedia Technol 12:180–185
20. Wang G-G, Guo L, Gandomi AH, Hao G-S, Wang H (2014) Chaotic Krill herd algorithm. Inf Sci. doi:10.1016/j.ins.2014.02.123
21. Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12:702–713
22. Du D, Simon D, Ergezer M (2009) Biogeography-based optimization combined with evolutionary strategy and immigration refusal. In: IEEE international conference on systems, man and cybernetics 2009, SMC 2009, pp 997–1002
23. Bhattacharya A, Chattopadhyay PK (2010) Hybrid differential evolution with biogeography-based optimization for solution of economic load dispatch. IEEE Trans Power Syst 25:1955–1964
24. Gong W, Cai Z, Ling CX (2010) DE/BBO: a hybrid differential evolution with biogeography-based optimization for global numerical optimization. Soft Comput 15:645–665
25. Ma H, Simon D (2011) Blended biogeography-based optimization for constrained optimization. Eng Appl Artif Intell 24:517–525
26. Boussaid I, Chatterjee A, Siarry P, Ahmed-Nacer M (2011) Hybridizing biogeography-based optimization with differential evolution for optimal power allocation in wireless sensor networks. IEEE Trans Veh Technol 60:2347–2353
27. Boussaïd I, Chatterjee A, Siarry P, Ahmed-Nacer M (2011) Two-stage update biogeography-based optimization using differential evolution algorithm (DBBO). Comput Oper Res 38:1188–1198
28. Boussaïd I, Chatterjee A, Siarry P, Ahmed-Nacer M (2012) Biogeography-based optimization for constrained optimization problems. Comput Oper Res 39:3293–3304
29. Chatterjee A, Siarry P, Nakib A, Blanc R (2012) An improved biogeography based optimization approach for segmentation of human head CT-scan images employing fuzzy entropy. Eng Appl Artif Intell 25:1698–1709
30. Saremi S, Mirjalili S (2013) Integrating chaos to biogeography-based optimization algorithm. Int J Comput Commun Eng 2:655–658
31. Mirjalili S, Mirjalili SM, Lewis A (2014) Let a biogeography-based optimizer train your multi-layer perceptron. Inf Sci 269:188–209. doi:10.1016/j.ins.2014.01.038
32. Gandomi A, Yang X-S, Talatahari S, Alavi A (2013) Firefly algorithm with chaos. Commun Nonlinear Sci Numer Simul 18:89–98
33. Digalakis J, Margaritis K (2001) On benchmarking functions for genetic algorithms. Int J Comput Math 77:481–506
34. Mirjalili S, Hashim SZM (2010) A new hybrid PSOGSA algorithm for function optimization. In: 2010 International conference on computer and information application (ICCIA), pp 374–377
35. Mirjalili S, Lewis A (2013) S-shaped versus V-shaped transfer functions for binary particle swarm optimization. Swarm Evol Comput 9:1–14
36. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
37. Mirjalili S, Mirjalili SM, Yang X-S (2013) Binary bat algorithm. Neural Comput Appl 1–19. doi:10.1007/s00521-013-1525-5
38. Mirjalili S, Hashim SM (2011) BMOA: binary magnetic optimization algorithm. In: 2011 3rd International conference on machine learning and computing (ICMLC 2011), Singapore, pp 201–206
39. 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 Evol Comput 1(1):3–18. doi:10.1016/j.swevo.2011.02.002
40. 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. J Heuristics 15:617–644
41. Wilcoxon F (1945) Individual comparisons by ranking methods. Biom Bull 1:80–83