S. Saremi, S. Mirjalili, A. Lewis, Biogeography-based optimisation with chaos, Neural Computing and Applications, Vol. 25, pp. 1077 - 1097, 2014, Springer DOI: http://dx.doi.org/10.1007/s00521-014-1597-x

Biogeography-based optimisation with chaos

Shahrzad Saremi, Seyedali Mirjalili, Andrew Lewis

School of Information and Communication Technology, Griffith University, Nathan, Brisbane, QLD 4111, Australia

shahrzad.saremi@griffithuni.edu.au, seyedali.mirjalili@griffithuni.edu.au, a.lewis@griffith.edu.au

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 10 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 (CBBO) algorithms are benchmarked on 10 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;

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 pre-defined number of generations. The evolution process is performed by selection, reproduction, mutation, and recombination operators. The EA algorithms 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 EA algorithms are mostly equipped with selection and recombination operators in order to explore search spaces. In contrast, the exploitation phase refers to the convergence toward 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 EA algorithms [13]. These issues leave EA prone to stagnation in local optima. In other words, EA algorithms 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 has 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.* developed chaotic crossover and mutation operators and proved that chaos theory can improve the performance of GA successfully [15]. 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 created the initial population for GA and DE by chaos theory and applied it to a water resource system problem [17]. 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 Krill Herd (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 relations 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.* combined ES with BBO to improve the local optima avoidance of BBO [22]. 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.* 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 [24]. In 2011, Ma and Simon proposed a blended migration operator and indicated that it could improve the exploration of the BBO algorithm [25]. 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 3 chaotic maps with BBO and tested it over 4 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 10 chaotic maps into this algorithm in order to extensively investigate the effectiveness of chaos theory for improving exploration and/or exploration 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: Section 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 Section 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 relations 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 toward 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 pre-defined 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:

$$\mu_k = \frac{E \times n}{N} \tag{2.1}$$

$$\lambda_k = I \times \frac{1-n}{N} \tag{2.2}$$

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.

Fig. 1 illustrates immigration and emigration rates. This figure shows that the probability of emigration is proportional to 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:

$$m_n = M \times \left(1 - \frac{p_n}{p_{max}}\right) \tag{2.3}$$

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

The pseudo code of the BBO algorithm is illustrated in Fig. 2.

In the next section the method of integrating the 10 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 10 different chaotic maps as shown in Table. 1 and Fig. 3.

It is worth mentioning here that Fig. 3 shows deterministic systems are also able to provide chaotic behaviours. There is no random component in Table. 1, but the chaotic behaviours of the equations is 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 depends on the range of chaotic map). However, it should be noted that the initial value may have significant impacts on the fluctuation pattern of some of the chaotic maps. 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 of 10 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:

```
if $C(t) < \lambda_i$ then
    Emigrate habitants from $H_i$ to $H_j$ chosen with the probability proportional to $\mu_i$        (3.1)
end if
```

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 equation (3.1) 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:

```
if $C(t) < \mu_i$ then
    select a random habitant in $x_i$ and replace it with $x_j$        (3.2)
end if
```

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

Equation (3.2) 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:

```
for i=1 to number of habitants at k-th habitat
    if C(t) < Mutation_rate(k) then
            Mutate i-th habitants                                  (3.3)
    end if
end for
```

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 (2.3) 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) are 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.

4. Experimental results and discussion

To evaluate the performance of the proposed CBBO algorithms, 10 standard benchmark functions are employed in this section [3, 33-38]. These benchmark functions are divided into two groups: multimodal and unimodal . The former group is suitable for benchmarking exploration, whereas the latter group is use to examine exploitation [36, 37, 39]. Table 2 lists these functions and their features, where *Dim* indicates 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.

Table 4 to Table 8 present the experimental results. The results are averaged over 10 independent runs. The average (*mean*) and standard deviation (*std*) of the best solution obtained in the last iteration are reflected in the tables. According to Derrac *et al.* [40], 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 [41] 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 [42], 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 less than 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 greater than 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.

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, and 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 greater than 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 Table 4, Table 5, and Table 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 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 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 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 < 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 greater than 0.5. This means that the Sinusoidal map provides less probability of selection, emigration, and mutation compared to others, from consideration of equation (3.1), equation (3.2), and equation (3.3). 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 less than 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 10 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 are better than other combinations investigated. This was due to the superior exploration of the CBBO with these two operators combined.

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] H. John, "Holland, Adaptation in natural and artificial systems," ed: MIT Press, Cambridge, MA, 1992.

[2] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential evolution*: Springer, 1997.

[3] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *Evolutionary Computation, IEEE Transactions on,* vol. 3, pp. 82-102, 1999.

[4] W. Banzhaf, J. Koza, C. Ryan, L. Spector, and C. Jacob, "Genetic programming," *Intelligent Systems and their Applications, IEEE,* vol. 15, pp. 74-84, 2000.

[5] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation,* vol. 17, pp. 4831-4845, 2012.

[6] L. Guo, G.-G. Wang, A. H. Gandomi, A. H. Alavi, and H. Duan, "A new improved krill herd algorithm for global numerical optimization," *Neurocomputing*.

[7] G. Wang, L. Guo, H. Wang, H. Duan, L. Liu, and J. Li, "Incorporating mutation scheme into krill herd algorithm for global numerical optimization," *Neural Computing and Applications,* pp. 1-19, 2012.

[8] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, "A chaotic particle-swarm krill herd algorithm for global numerical optimization," *Kybernetes,* vol. 42, pp. 9-9, 2013.

[9] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, "An effective krill herd algorithm with migration operator in biogeography-based optimization," *Applied Mathematical Modelling,* 2013.

[10] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, "Stud krill herd algorithm," *Neurocomputing,* vol. 128, pp. 363-370, 2014.

[11] G.-G. Wang, A. H. Gandomi, A. H. Alavi, and G.-S. Hao, "Hybrid krill herd algorithm with differential evolution for global numerical optimization," *Neural Computing and Applications,* pp. 1-12, 2013.

[12] S. Mirjalili, S. Z. Mohd Hashim, and H. Moradian Sardroudi, "Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm," *Applied Mathematics and Computation,* vol. 218, pp. 11125-11137, 2012.

[13] S. Mirjalili, A. Lewis, and S. M. Mirjalili, "Adaptive gbest-guided gravitational search algorithm," *Neural Computing and Applications,* p. In press, 2014.

[14] S. H. Kellert, *In the wake of chaos: Unpredictable order in dynamical systems*: University of Chicago Press, 1993.

[15] N. Wang, L. Liu, and L. Liu, "Genetic algorithm in chaos," *OR Transactions,* vol. 5, pp. 1-10, 2001.

[16] L.-J. Yang and T.-L. Chen, "Application of chaos in genetic algorithms," *Communications in Theoretical Physics,* vol. 38, pp. 168-172, 2002.

[17] V. Jothiprakash and R. Arunkumar, "Optimization of Hydropower Reservoir Using Evolutionary Algorithms Coupled with Chaos," *Water Resources Management,* pp. 1-17, 2013.

[18] G. Zhenyu, C. Bo, Y. Min, and C. Binggang, "Self-adaptive chaos differential evolution," *Advances in natural computation,* pp. 972-975, 2006.

[19] S. Saremi, S. M. Mirjalili, and S. Mirjalili, "Chaotic Krill Herd Optimization Algorithm," *Procedia Technology,* vol. 12, pp. 180-185, 2014.

[20] G.-G. Wang, L. Guo, A. H. Gandomi, G.-S. Hao, and H. Wang, "Chaotic Krill Herd algorithm," *Information Sciences,* 2014.

[21] D. Simon, "Biogeography-based optimization," *Evolutionary Computation, IEEE Transactions on,* vol. 12, pp. 702-713, 2008.

[22] D. Du, D. Simon, and M. Ergezer, "Biogeography-based optimization combined with evolutionary strategy and immigration refusal," in *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, 2009, pp. 997-1002.

[23] A. Bhattacharya and P. K. Chattopadhyay, "Hybrid differential evolution with biogeography-based optimization for solution of economic load dispatch," *Power systems, IEEE Transactions on,* vol. 25, pp. 1955-1964, 2010.

[24] W. Gong, Z. Cai, and C. X. Ling, "DE/BBO: a hybrid differential evolution with biogeography-based optimization for global numerical optimization," *Soft Computing,* vol. 15, pp. 645-665, 2010.

[25] H. Ma and D. Simon, "Blended biogeography-based optimization for constrained optimization," *Engineering Applications of Artificial Intelligence,* vol. 24, pp. 517-525, 2011.

[26] I. Boussaid, A. Chatterjee, P. Siarry, and M. Ahmed-Nacer, "Hybridizing biogeography-based optimization with differential evolution for optimal power allocation in wireless sensor networks," *Vehicular Technology, IEEE Transactions on,* vol. 60, pp. 2347-2353, 2011.

[27] I. Boussaïd, A. Chatterjee, P. Siarry, and M. Ahmed-Nacer, "Two-stage update biogeography-based optimization using differential evolution algorithm (DBBO)," *Computers & Operations Research,* vol. 38, pp. 1188-1198, 2011.

[28] I. Boussaïd, A. Chatterjee, P. Siarry, and M. Ahmed-Nacer, "Biogeography-based optimization for constrained optimization problems," *Computers & Operations Research,* vol. 39, pp. 3293-3304, 2012.

[29] A. Chatterjee, P. Siarry, A. Nakib, and R. Blanc, "An improved biogeography based optimization approach for segmentation of human head CT-scan images employing fuzzy entropy," *Engineering Applications of Artificial Intelligence,* vol. 25, pp. 1698-1709, 2012.

[30] S. Saremi and S. Mirjalili, "Integrating Chaos to Biogeography-Based Optimization Algorithm," *International Journal of Computer and Communication Engineering,* vol. 2, pp. 655-658, 2013.

[31] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Let a biogeography-based optimizer train your Multi-Layer Perceptron," *Information Sciences,* 2014.

[32] A. Gandomi, X.-S. Yang, S. Talatahari, and A. Alavi, "Firefly algorithm with chaos," *Communications in Nonlinear Science and Numerical Simulation,* vol. 18, pp. 89-98, 2013.

[33] J. Digalakis and K. Margaritis, "On benchmarking functions for genetic algorithms," *International journal of computer mathematics,* vol. 77, pp. 481-506, 2001.

[34] S. Mirjalili and S. Z. M. Hashim, "A new hybrid PSOGSA algorithm for function optimization," 2010, pp. 374-377.

[35] S. Mirjalili and A. Lewis, "S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization," *Swarm and Evolutionary Computation,* vol. 9, pp. 1-14, 2013.

[36] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Advances in Engineering Software,* vol. 69, pp. 46-61, 2014.

[37] S. Mirjalili, S. M. Mirjalili, and X.-S. Yang, "Binary bat algorithm," *Neural Computing and Applications,* pp. 1-19, 2013.

[38] S. Mirjalili and S. M. Hashim, "BMOA: binary magnetic optimization algorithm," in *2011 3rd international conference on machine learning and computing (ICMLC 2011), Singapore*, 2011, pp. 201-206.

[39] S. Mirjalili and A. Lewis, "S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization," *Swarm and Evolutionary Computation,* vol. 9, pp. 1-14, 2013.

[40] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation,* 2011.

[41] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization," *Journal of Heuristics,* vol. 15, pp. 617-644, 2009.

[42] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin,* vol. 1, pp. 80-83, 1945.