British Journal of Mathematics & Computer Science
10(1): 1-12, 2015, Article no.BJMCS.17145
ISSN: 2231-0851

SCIENCEDOMAIN *international*
www.sciencedomain.org

Numerical Function Optimization Solutions Using the African Buffalo Optimization Algorithm (ABO)

Julius Beneoluchi Odili[1*] and Mohd Nizam Mohmad Kahar[1]

[1]*Faculty of Computer Systems and Software Engineering, Universiti Malaysia Pahang, Kuantan, Malaysia.*

Article Information

DOI: 10.9734/BJMCS/2015/17145
Editor(s):
(1) Doina Bein, Applied Research Laboratory, The Pennsylvania State University, USA.
Reviewers:
(1) G. Y. Sheu, Chang-Jung Christian University, Taiwan.
(2) Anonymous, MM University, Haryana, India.
(3) Anonymous, Selcuk University, Turkey.
Complete Peer review History: http://sciencedomain.org/review-history/9916

Original Research Article

Received: 27 February 2015
Accepted: 21 March 2015
Published: 27 June 2015

Abstract

This paper proposes a new meta-heuristic approach to solving continuous optimization problems using 21 benchmark test cases. The African buffalo algorithm evolved from an understanding of this animal's survival instincts and the search techniques they utilize in the African forests and savannahs. The African buffalo employs its exceptionally intelligent, cooperative and democratic attitude in its search for the optimal path to pasture. This enables it to get results faster than some other search agents. The African Buffalo Optimization (A.B.O) algorithm simulates the African buffalos' behaviour by encapsulation in a mathematical model; which solves a number of continuous optimization problems. When compared to the Genetic Algorithm (GA),Chaotic Gray-coded Genetic Algorithm and the Improved Genetic Algorithm (IGA), the results obtained from African Buffalo Optimization show that the algorithm works well and can be extended to solving other optimization problems like: path planning, scheduling, vehicle routing.

Keywords: Numerical function optimization; African buffalo optimization (ABO); global optimization; multimodal; uni-modal.

1 Introduction

The never-ending demands for better ways of solving problems and cheaper ways of getting things done has led many researchers to investigate the field of optimization leading to the discovery of several optimization algorithms such as Ant Colony Optimization [1], Artificial Bee Colony [2], Genetic Algorithm [3], Particle Swarm Optimization [4], Differential Evolution (DE) [5] amongst many others. These techniques have been successfully applied to solve a number of problems in a few minutes: problems that was supposed to take

**Corresponding author: odili_julest@yahoo.com;*

several months or even years. Some of these hitherto thought- to- be very difficult and time consuming problems include the travelling salesman's problem [2], job scheduling [3] and vehicle routing [3].

However, some of these algorithms have their drawbacks leading to scientific investigation with a view to solving the limitations in the popular algorithms. Some of the drawbacks of the above-listed algorithms include premature convergence [5] to complicated fitness function [2], inefficiency in the exploration of the search space [2], the use of several parameters [3], weakness in refining the search space at a later stage [4] and complex implementation strategies [6]. Some of these weaknesses informed the development of the African Buffalo Optimization algorithm. This algorithm draws its inspiration from observing a specie of African wild cows called African Buffalos in their quest for grazing pastures in the African forests. This animal is in competition with other herbivorous animals which most times require less intake of pastures than this large animal with big appetite. A lot of ingenuity is required if she is to survive the competition and sometimes the hostility of African lions and human hunters. This work is an attempt to model this animal's ingenuity in navigating her way through several thousands of kilometers in the vast African forests with the sole aim of tracking the wet seasons in different locations where it could satisfy its appetite.

African Buffalo Optimization (ABO) is an attempt to develop a user-friendly, robust, effective, efficient yet simple-to-implement algorithm that will demonstrate exceptional capacity in the exploitation and exploration of the search space. ABO attempts to solve the problem of pre-mature convergence or stagnation by ensuring that the location of the each buffalo is regularly updated in relation to the animal's best previous location and the present location of the best buffalo location in the herd. In a situation, for instance, where the leading (best) buffalo's location is not improved in a number of iterations, the entire herd is re-initialized. Tracking the best position and speed of each buffalo ensures adequate exploitation of the search space and tapping into the experience of other buffalos as well as that of the best buffalo enables the ABO to achieve adequate exploration. Similarly ABO ensures fast convergence with its use of very few parameters, primarily the acceleration constants lp1 and lp2. These constants enable the movement of the animals towards greater exploitation or exploration depending on the focus of the algorithm at a given iteration.

The first section of this paper highlights the motivation for this research, the second introduces the African Buffalo Optimization (ABO) algorithm. This is followed by an explanation of the basic flow of the algorithm indicating the general working and movements of the buffalos in search for a solution. The third section highlights the Genetic Algorithm, Improved Genetic Algorithm and the Chaotic Gray-coded Genetic Algorithm. The fourth section of the paper discusses the methodology used in the and the fifth section presents the experimental results as well as a detailed analysis of the results obtained. This is followed by the Conclusion, recommendation for future research endeavours with the new algorithm references and Appendix.

2 Introducing the African Buffalo Optimization (ABO)

African Buffalo Optimization (ABO) simulates the alert ('maaa') and alarm ('waaa') behavior of African buffalos in its foraging ventures. These are the two basic sounds of the African Buffalos with which they are able to organize themselves to search for food and defend themselves whenever they are attacked. The waaa soundis used to mobilize the buffalos to move on to explore the search space while the maaa sound tells the buffalos to stay on to exploit their environment since it is safe and has sufficient pastures. With these sounds, the buffalos are able to optimize their search for food source. The ABO is a population-based algorithm in which individual buffalos work together to solve a given problem. Each buffalo within the ABO algorithm represents an aggregate object containing a number of elements. The African Buffalo Optimization algorithm recognises the democratic nature of the buffalos and incorporates this into the algorithm. This is represented by Equation (1). In this equation, there is an interaction of the various buffalos with the leading (best) buffalo at a particular iteration, a comparison with each buffalos best ever location in relation to the target solution as well as a memory of each animals immediate past location. The interaction of these three elements leads to the waaa equation (2) where the animals move on to explore other locations depending on the result of the democratic equation (1).

2.1 The Working of the ABO Algorithm

The algorithm begins by initializing the population of animals through conscious allocation of random locations to each buffalo within the N-dimensional space. After this, the algorithm updates each buffalo's fitness and determines the bp_{max} (individual buffalo's personal location) and bg_{max} (the herd's best location) in a particular iteration in relation to the optimal solution. If the present fitness is better than the individual's maximum fitness (bp_{max}), it saves the location vector for the particular buffalo. If the fitness is better than the herd's overall maximum, it saves it as the herd's maximum (bg_{max}). Finally it updates the buffalos location and looks at the next buffalo in the population. At this point, if our global best fitness meets our exit criteria, it ends the run and provides the location vector as the solution to the given problem. Each element of the solution vector represents the independent variable of the given problem. The ABO algorithm is shown is Fig. 1.

1.Objective function $f(\mathbf{x})$ $\mathbf{x} = (x_1, x_2, ..., x_n)^T$

2. Initialization: randomly place buffalos to nodes at the solution space;

3. Update the buffalos fitness values using Equation (1)

$$w_{.k+1} = w_{.k} + lp_1 r_1(bg_{max} - m_{.k}) + lp_2 r_2(bp_{max.k} - m_{.k}) \quad (1)$$

where $w_{.k}$ and $m_{.k}$ represents the exploration and exploitation moves respectively of the kth buffalo (k=1,2,..........N) ; lp_1 and lp_2 are learning factors; r_1 and r_2 are random numbersbetween [0,1];

bg_{max} is the herd's best fitness and bp_{max}, the individual buffalo's best

4. Update the location of buffalo k ($bp_{max.k}$ and $bg_{max.k}$) using (2)

$$. m_{.k+1} = \lambda (w_{.k} + m_{.k}). \quad (2)$$

λ is a unit of time

5. Is bg_{max} updating. Yes, go to 6. No, go to 2

6. If the stopping criteria is not met, go back to algorithm step 3

7. Output best solution.

Fig. 1. The ABO algorithm

A close look at the ABO algorithm in Fig. 1, shows that the algorithm's democratic equation (1) has three parts, namely the first $w_{.k}$ represents the memory of the buffalos past location. This is a list of solutions that can be used as an alternative for the current local maximum location. There is a probability of choosing one of the target lists of solutions of the buffalo's memory instead of the present herd's maximum point. The second part $lp_1 r_2$ $(bg_{axax.} - m_{.k})$ indicates the cooperative and information-sharing part of the algorithm allowing each buffalo to tap into the experience of the entire herd in its search efforts and third *lp2r2* (*bpmax.k - m.k*) displays the intelligence of the animals and is a pointer to the buffalo's personal experience.

2.2 Controlling the Movement of the Buffalos

Two main equations control the movement of buffalos within the search space and these are Equations (1) and (2) (refer Fig. 1). This democratic equation (1) provides for the decision to either stay on to exploit the environment further or to move on to explore other areas based on the collective intelligence of the herd after interacting with each other in a communal decision-making process [7]. The maaa equation (2) propels the buffalos to move on to explore other areas based on the outcome of democratic equation (1) after due consideration of the two competing forces (bp_{max} *and* bg_{max}). The λ parameter which defines the discrete time interval over which the buffalo must move is usually set to 1.0. The application of these two Equations results in a new location for the buffalos.

It should be observed that equation (1) aside from the memory part ($w_{.k}$), has two other controlling features, namely, the global maximum ($bg_{axax.}$) and the personal maximum positions (bp_{max}): each defining the representative influence over the animal's location. The algorithm subtracts the dimensional element $m_{.k}$ from the maximum vector and then multiplies this by a random number $(r_{1,} r_2)$ usually between 0.0 to 0.6 and a learning/acceleration parameter $(lp_{1,} lp_2)$. Using the random numbers between 0.0 to 0.6 has so far proved effective in obtaining fast convergence. Further research is on-going to get figures that may yield better results. The sum of these products is then added to the speed for the given dimension of the sector. This procedure is performed for each element of the speed vector. It should be emphasized that the random numbers give an amount of randomness in the path to help the animals move throughout the search space. It does this by randomly giving more or less emphasis to the global (bg_{max}) or personal maximum solutions depending on the need for more exploration or exploitation respectively as the algorithm progresses.

2.3 African Buffalo Optimization (ABO) for Global Optimization

Modern day problems in engineering, social science, business, medicine, and applied sciences are getting more complex. In mathematical term, these problems are no longer linear, quadratic or mono-modal but are now multimodal. The domains of the problems cum their objective functions are often multimodal with peaks, valleys, channels, and flat hyper-planes of different heights. Solving these types of problems, which are classified as global optimization problems, to their optimal solutions has become a true challenge [8].

In order to explore the potentials of the proposed algorithm-the African Buffalo Optimization- it is necessary to investigate her capacity to search the different types of solution spaces ranging from the mono-modal to multi-modal, constrained to unconstrained, separable and non-separable solution spaces [9,10]. A mono-modal function $f(x)$ has a single extreme, that is to say, it has a single minimum or a single maximum within the range that is specified for x. Similarly, a function is said to be multimodal if such a function has more than one peak, either on the minimum or the minimum sides. Furthermore a function is said to be separable if it can be written as a sum of 'p' functions of just one variable. Functions that are non-separable are more difficult to optimize. This is due to the fact that accurate search directions depend or two or more elements within the search space (or solution vector). The situation gets messier in a case of a multi-modal function [11,12]. The last issue that poses a problem to optimization algorithms is the case of multiple dimensions. This is because the number of local optima increases with the increase in problem dimensions [13].

Such benchmark multimodal functions test the capacity of algorithms to escape local minima or extreme. If the exploration capacity of an algorithm is below par, such algorithm will be stacked in some local minima. Among the functions, we shall investigate includes those that have flat surfaces. Such functions pose some challenges to algorithms as they provide insufficient information to enhance the search [14].

3 Review of Other Similar Algorithms

3.1 Chaotic Gray-coded Genetic Algorithm

In order to reduce the computational amount and improve computational precision for nonlinear optimizations, Xiahua Yang et al. [15] developed the Chaos Gray-coded Genetic Algorithm (CGGA), in which initial population are generated by chaos mapping with the aid of newer chaos mutation and Hooke–Jeeves evolution operators As the search range shrinks, the CGGA slowly traces the optimal result through the excellent individuals obtained through the algorithm until convergence.

3.2 Improved Genetic Algorithm

Concerned by what they perceived as the weaknesses of the Genetic Algorithm, J. Andre, P. Siarry and T Dognon proposed what they called Improved Genetic Algorithm [16]. The main aim of this algorithm is to enhance the working of the standard Genetic Algorithm resulting from the refinement of the evolution processes. In validating their algorithm, it was tested on a number of benchmark numerical functions.

3.3 Genetic Algorithm

Genetic Algorithm imitates the process of natural selection, evolution, inheritance, cross and mutation to arrive at solutions to optimization problems [17]. It belongs to the group of algorithms called Evolutionary Algorithms (EA).

3.4 Experiments Benchmark Functions

For the purpose of this study, a total of 20 benchmark functions were investigated. The benchmark functions investigated in this study includes: f_1, f_3, f_{5n}, f_{10n}, f_{15n}, Branin, Gold Price, Hosc45, Brown1,Camelback6, Shubert, Hartman1, Hartman2, PShubert1, PShubert2, Brown1, Shekel2, Shekel3, Brown3, Quartic and Rosenbrock [13,18]. A brief description of each of the twenty-one benchmark test functions may be helpful in order to enable us appreciate them better. This is done in the Appendix by the end of this paper.

4 Methodology

The effectiveness of an algorithm is measured by its ability to be efficient in searching the solution space, speed, convergence rate, and robustness [17]. This paper, in testing the robustness, therefore, examines the capacity of the African Buffalo Optimization (ABO) to search diverse search spaces ranging from mono-modal to multi-modal, separable to non-separable search spaces. In testing for speed, we consider the average number of functional evaluation required to obtain the optimal solution vis-à-vis that of other algorithms. Similarly, the convergence rate is tested by comparing the optimal solution obtained by the ABO with the benchmark result of the different global optimization test functions with the aim of bring out the Relative Error percentage. A comparative lower relative error is a hallmark of good algorithm. This is calculated using the formula:

$$\text{Relative error } (\%) = \frac{|Opt_{ABO} - Opt_R|}{Opt_R} \times 100\ (\%) \tag{3}$$

Where relative error is set by the optimum error, Opt_{ABO} is the optimal result obtained by ABO after 10 runs, and Opt_R is the optimum result of the benchmark optimization functions. However, in a situation where the Opt_R is 0, we will simply use the equation:

$$\text{Relative error } (\%) = |Opt_{ABO} - Opt_R| \times 100\ (\%) \tag{4}$$

Also included in this work is the Success Rate of the ABO which is a measure of how many times (in percentage) the algorithm is able to obtain the optimum result for each test function.

5 Experiment and Results

The experiments were performed using a desktop computer running Windows 7, 64-bit Operating System, Intel Core [TM], i7-3770 CPU@ 3.4GHz, 3.4GHz, 4GB RAM. These benchmark function equations were coded in MATLAB programming language and were run using MATLAB 2012b tool. The data obtained from the experiment with the ABO was compared with results obtained from similar experiments using the Genetic Algorithm (GA) and the Improved Genetic Algorithm (IGA). Comparative data are obtained from [16,15].

In Table 1, N = number of variables required by each test function, E (%) = Relative Error, Opt = benchmark Optimal Result obtained by the particular algorithm, AFE= Average number of Function Evaluation, S (%) = percentage Success, UN = Uni-modal and Non Separable, MS= Multimodal and Separable, US=Uni-modal and Separable.

Table 1. ABO experimental results comparison with GA and IGA

S/N	Function name	Opt $_R$	N	Xte	ABO				GA				IGA				CGGA			
					Opt	E (%)	AFE	S (%)	Opt	E (%)	AFE	S (%)	Opt	E (%)	AFE	S (%)	Opt	E (%)	AFE	S (%)
f_1	Branin	0.3978	2	MN	0.3978	0	152	100	0.3978	0	8125	81	0.3978	0	2040	100	0.3978	0	300	100
f_2	Brown1	2	20	UN	2.2725	6.8	8039	87	43.6281	861.2	6844	0	8.5516	327.5	12864	0	1.9987	0.065	319	100
f_3	Brown3	0	20	UN	0.0010	1	4000	99	1.3060	13	8410	5	0.6746	67.5	106857	5	0.06952	69.5	70129	82
f_4	Camelback6	-1.03163	2	MN	-1.0316	0	582	100	-1.0316	0	1316	98	-1.3163	0	2040	98	-1.0316	0	301	100
f_5	Gold Price	3	2	MN	3.0000	0	103	100	3.0000	0	8185	59	3.003	0.003	1316	100	3.0008	0.0003	305	100
f_6	Hartman3	-3.8627	3	MN	-3.8627	0	91	100	-3.8625	0.000	1993	94	-3.8611	0.04	1680	100	-3.8677	0.12	337	100
f_7	Hartman6	-3.3223	6	MN	-3.3223	0	231	100	3.3065	4.7	19452	23	-3.3138	0.2	53792	100	-3.3195	0.008	400	100
f_8	Quartic	-0.3523	2	US	0.3523	0	116	100	0.3523	0	8181	83	.0.352	0	3168	100	-0.3523	0	308	100
f_9	Hosc45	1	10		2.0000	100	145	50	1.9951	99.9	11140	0	1.0094	9	126139	92	1.0000	0	307	100
f_{10}	Rosenbrock	0	2	UN	0.0000	0	127	100	7.39	-	-	-	-	-	-	-	-	-	-	-
f_{11}	Shekel5	-10.1532	4	MN	-10.1532	0	150	100	-10.1349	1.7	7495	1	-10.149	0.4	36388	97	-10.1520	0.011	600	100
f_{12}	Shekel7	-10.4029	4	MN	-10.4029	0	228	100	-10.1677	2.3	8452	0	-10.383	1.2	36774	08	-10.3873	1.5	600	100
f_{13}	Shekel10	-10.5364	4	MN	-10.5364	0	280	100	-104034	0.1.5	8521	0	-10.514	0.21	36772	100	-10.5208	1.5	719	100
f_{14}	Shubert	-186.7309	2	MS	-186.7309	0	33	100	-186.7310	0.0005	6976	93	-186.73	0.002	2364	100	-186.7309	0	359	100
f_{15}	F1	-1.2323	1	MS	-1.2323	0	44	100	-1.2323	0	5566	100	-1.2323	0	784	100	-1.2323	0	300	100
f_{16}	F3	-12.0313	1	MS	-12.0313	0	46	100	-12.0312	0.0009	5347	100	-12.031	0	744	100	-12.3123	2.3	300	100
f_{17}	F5n	0	20	MS	0.0140	1.4	763	87	0.4735	47.3	8725	0	0.0001	0.01	99945	100	0.0000	0	867	100
f_{18}	F10n	0	20	MS	0.0000	0	720	100	7.8352	783.5	9298	0	0.0001	0.01	11392	49	0.0000	0	1860	100
f_{19}	F15n	0	20	MS	0.0000	0	794	100	0.5212	52.1	9541	0	0.0003	0.03	102413	100	0.0000	0	859	100
f_{20}	PShubert1	-186.7309	2	MS	-185.049	0.9	33	0	-186.7300	4.8	7192	63	-186.69	0.03	8853	100	-186.7309	0	509	100
f_{21}	PShubert2	-186.7309	2	MS	-185.226	0.8	33	0	-186.7310	0.54	7303	59	-186.71	0.015	4116	100	-186.7309	0	429	100

S/N= Serial number; N= Number of variables; Xte= Characteristics; Opt$_R$= Optimal result obtained by a given algorithm; E (%) = Relative error percentage; AFE = Average number of function evaluation; S (%) = Percentage of successful runs

The table above shows the capacity of the ABO to search different search spaces at a competitive rate of performance. Using the Global Relative Error percentage rate as a benchmark, it is obvious that ABO performed well in relation to the other the other three algorithms. While the Global Error percentage rate of GA is 93%, that of IGA is 3.9%, CCGA is 0.8% and ABO is 5.6%. This is obtained by simply obtaining the mean of all Relative Error Percentage. Similarly, it should be observed that ABO obtained the global optimal in 15(f_1,f_4,f_5,f_6, f_7,f_8, f_{10-16}, f_{18}, f_{19})and 99% accuracy in other 3(f_3,f_{20},f_{21}) test cases out of the 21 test functions, GA obtained the optimal result in 5(f_1, f_4, f_5, f_8, f_{15})of the test cases and very near optima in other 4(f_6, f_{13}, f_{16}, f_{21})cases obtaining about 99%. The CCGA obtained the optimal result in 11 benchmark cases (f_1, f_4, f_8, f_9, f_{14}, f_{17-21}) and 99% accuracy in four other functions (f_2, f_5,f_7, f_{11}).In total, GA obtained the optimal result in 10 test functions. The IGA obtained above 99% in 16 (f_1, f_4, f_5, f_6, f_7, f_{11}, f_{12}, f_{13}, f_{14}, f_{15},f_{16},f_{17},f_{18}, f_{19}, f_{20}, f_{21}) out of the 21 test functions. From this analysis, ABO slightly outperformed the IGA with 18 optimal or near optimal results to CGA's 15 and IGA's 16. The least performer here is the GA which was able to obtain the 99% optimal result in nine cases. This good performance of the ABO is traceable to its ability to search both locally and globally at the same time using the path-relinking technique [18]. Another factor for its success is the excellent information exchange between the best buffalo bg_{max} with other buffalos during the search process.

In comparing the three best performing algorithm in Table 1, that is the ABO, CCGA and IGA one fact emerged, i.e., IGA performs better in Multimodal search spaces more than in Uni-modal space. A look at the Table reveals that IGA's and CCGA's worst performance is in $f_3$where they obtained Relative Error of 67.7% and 69.5% respectively: that is a Uni-modal environment. Same applies to f_9, whereas it best performance are in f_1,f_4, f_7, f_{15},f_{16}: all multi-modal search environments except Quartic. The ABO, however is more robust as it is able to obtain optimal result in 15 test cases as pointed out earlier in different search environments ranging from Multi-modal Non Separable cases to Multimodal Separable, Uni-modal, Non-Separable to Uni-modal, Separable. This is a mark of good algorithm performance. Similarly, though the GA was unable to obtain optimal results in some of the test cases, its robustness is encouraging. It obtained results in varying search spaces.

Another interesting discovery is the correlation between the performances of the algorithms in relation to the number of variables. It is obvious that the algorithms performed better in test functions that have very few variables. For instance all the algorithms obtained optimal result in f_{15} and f_{16} which has both one variable each. They follow the same trend in their performance in numerical test functions that have two variables each. This can be seen in their excellent or near-excellent performances in f_1, f_4, f_5, f_8, f_{14}, f_{20} and f_{21}. In the same vein, the posed a good performance in f_6 that uses three variables: f_{11}, f_{12}, and f_{13} that use four variables and f_7 that uses six variables. From test functions that use up to 10 variables or more, they began experiencing varying levels of difficulties in obtaining the optimal result. Of particular interest is $f_2$and f_3 where the algorithms had their worst performances. The cases in f_{17}, f_{18} and f_{19} are not much different. The only slight exception here is CGGA. Based on this observation, it could be safely said that the more the number of variables required to solve a problem, the more difficult it is for the algorithms to obtain optimal solutions.

Again, let us look at the speed at which these algorithms obtain results. Since speed is a function of the programming expertise of the programmer, programming language as well as the hardware configuration [8,17], the speed evaluation shall be done through the assessment of the Average Functional Evaluation (AFE) of the different algorithms in obtaining optimal result. A close look at Table shows clearly that ABO is the fastest in virtually all test cases. Another fast performer is the CCGA which was faster than the ABO in two test cases (f_2, f_4). The speed of ABO is significant because cost is a function of speed and since the ABO is faster, it is more cost effective. The speed of ABO is traceable to its simplicity and use of very few parameters in its search process. This way it utilizes less CPU resources.

Finally, in evaluating the efficiency of the algorithms, we assess the Success Rate of each algorithm and compare with the others. Here, the ABO performed well achieving 100 % success in 15 of the 21 test cases, the IGA obtained 100% success in 13 and the least performer is GA 100% success rate in two cases only. The best performer judging by this criterion is the CGGA which scored 100% in all cases except one.

However, the data in Table shows that IGA is another consistent performer securing 0% and 5% just in two cases with the ABO securing 0% in two cases compared to the GA's 0% in six cases.

6 Conclusion

This study proposed a novel algorithm called the African Buffalo Optimization (ABO) and used it to determine the global optimal results of 21 benchmark numerical function optimization problems ranging from multimodal to uni-modal, separable to non-separable search spaces. The results obtained was used to compare the results from Genetic Algorithm (GA) and the Improved Genetic Algorithm (IGA). The outcomes displays the excellent performance of the ABO. More the consistency of the ABO performance attests to its robustness as can be seen from the test cases. It is, therefore, safe to conclude that the ABO is a worthy addition to the ever-growing number of optimization algorithms

7 Future Work

Since this work covered only 21 test functions, it is the authors desire that this algorithm should be used to test other benchmark functions and comparisons of such results made with other algorithms not covered in this study.

Acknowledgments

The authors acknowledge the Faculty of Computer Systems and Software Engineering of the Universiti Malaysia Pahang, Gambang, Kuantan, Malaysia for providing the conducive environment and funding of this research.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Reinelt G. TSPLIB - A travelling salesman problem library. ORSA Journal on Computing. 1991;3(4) Fall:376-384

[2] Dorigo M, Stützle Ant T. Colony optimization. Cambridge: The MIT Press; 1995.

[3] Bin Y, Zhong. Zhen Y, Baozhen Y. An improved ant colony optimization for vehicle routing. European Journal of Operational Research. 2009;196:171–176.

[4] Angeline P. Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. Evolutionary Programming VII. Springer, Berlin. 1998;601–610.

[5] Mezura-Montes E, Velázquez-Reyes J, Coello CA. A comparative study of differential evolution variants for global optimization. Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation. ACM Press, New York. 2006;485–492.

[6] Karaboga D, Akay B. A modified artificial bee colony (ABC) algorithm for constrained optimization problems. Apple Soft Computing. 2011;11(3):3021–3031.

[7] Wilson DS. Altruism and organism: Disentangling the themes of multi-level selection theory. The American Naturalist. 1997;150 (S1):122-134.

[8] Ali MM, Khompatraporn C, Zambinsky C. A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. Journal of Global Optimization; 2005.

[9] Shi Y, Eberhart R. Parameter selection in particle swarm optimization. Evolutionary Programming VII. Springer, Heidelberg. 1998;591–600.

[10] Khaze SR, Maleki I, Hojjatkhah S, Bagherinia A. Evaluation of the efficiency of artificial bee colony and the firefly algorithm in solving the continuous optimization. International Journal on Computational Science & Applications (IJCSA). 2013;3:23-35.

[11] Xin-She Y. Engineering optimization: An introduction with metaheuristc applications. New Jersey, USA: John Wiley and Sons Inc; 2010.

[12] Xin-She Y, et al (Eds.). Swarm intelligence and bio-inspired computation. Amsterdam: Elsevier Inc; 2013.

[13] Zambrano-Bigiarini. Test Functions For Global Optimization; 2005.
Available: http://www.rforge.net/doc/packages/hydroPSO/test_functions.html

[14] Machado JM, Yang S, Ho SL, Ni P. A common tabu search algorithm for the global optimization of engineering problems. Comput. Methods Appl. Mech. Engr. 2001;190:3501–3510.

[15] Yang X, Yang Z, Yin XJ. Chaos gray-coded genetic algorithm and its application for pollution source identifications in convection–diffusion equation. Communications in Nonlinear Science and Numerical Simulation. 2006;13(8):1676–1688.

[16] Andre J. Dognon T, Siarry P. An improvement of the standard genetic agorithm fighting premature convergence in continuous optimization. Advances in Software Engineering. (Elsevier Inc.). 2001;32: 41-60.

[17] Stochastic global optimization algorithms workshop. University of Canterbury. Christchurch, New Zealand, June; 2005.

[18] Glover F, Laguna M, Martí R. Fundamentals of scatter search and path relinking. Control and Cybernetics. 2000;29(3):653-684.

APPENDIX

[1.] Branin RCOS Function. (Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal)

$$f(x) = \left(\frac{x_2 - 5.1x_1^2}{4\pi^2} + \frac{5x_1}{\pi} - 6\right)^2 + 10\left(1 - \frac{\pi}{8}\right) + 10$$

with domain $-5 \le x_1 \le 10$, $0 \le x_1 \le 15$. It has three global minima at $(x_1, x_2) = (\{-\pi, 12.275\}, \{\pi, 2.275\}, \{3\pi, 2.425\})$, $f(x) = 0.3978873$.

[2.] Brown1 (20 variables): $f(x) = \sum_{i\in J}(x_i-3)^2 + \sum_{i\in J}(10^{-3}(x_i-3))^2 - (x_i - x_i + 1) + e_{20}(x_i - x_i + 1)$ where $J = \{1, 3, \ldots, 19\}$, $-1 \le x_i \le 4$ for $1 \le i \le 20$ and $x = [x_1, \ldots, x_{20}]T$. . The global minimum is located at $x_i = 0$, $f(x) = 2$.

[3.] Brown3 Function. (Continuous, Differentiable, Non-Separable, Scalable, mono-modal) f (x) $=\sum_{i=1}^{n-1}(x_i^2)(x_i+1)+1)(x_i^2+1)$ subject to $-1 \le x_i \le 4$. The global minimum is located at $\mathbf{x}_i = 0$, f = 0.

[4.] Camel Function – Six Hump (Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal) $f_{30}(x) = (4-2.1x_i^2 + x_1^4/1/3)x_1^2 + x_1x_2 + (4x_2^2 - 4)\, x_2^2$ subject to $-5 \le x_i \le 5$. The two global minima are located at $(x_1, x_2) = (\{-0.0898, 0.7126\}, \{0.0898, -0.7126, 0\})$, $f = -1.0316$.

[5.] Goldstein Price Function (Continuous, Differentiable, Non-separable, Non-Scalable, Multimodal) $f = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_{21} - 14x_2 + 6x_1x_2 + 3x_2)] \times [30 + (2x_1 - 3x_2)2(18 - 32x_1 + 12x_{21} + 48x_2 - 36x_1x_2 + 27x_2)]$ subject to $-2 \le x_i \le 2$. The global minimum is located at $x_i = f(0,-1)$, f = 3.

[6.] Hartman 3 Function (Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal) $f(x) = \sum_{i=1}^{4} c_i \exp\,(-\sum_{j=1}^{3} aij\ (xj - pi)^2$

subject to $0 \le xj \le 1$, $j \in \{1, 2, 3\}$ with constants Aij , pij and ci are given as

A = Aij =

(3	10	30
0.1	10	35
3	10	30
0.1	10	35)

P= Pi =

(0.3689	0.1170	0.2673
0.4699	0.4837	0.7470
0.1091	0.8732	0.5547
0.03815	0.5743	0.8828)

The global minimum is located at x= f(0.1140, 0.556, 0.852), $f(x) \approx -3.862782$

[7.] Hartman 6 Function. (Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal) $f(x) = \sum_{i=1}^{4} ci \exp \quad (-\sum^2$

subject to $0 \le xj \le 1$, $j \in \{1, \cdots, 6\}$ with constants aij, pij and ci are given as

(10	3	17	3.8	1.7	8		(1
0.05	10	17	0.1	8	14	c=ci=	1.2
3	3.5	1.7	10	17	8		3
17	8	0.05	10	0.1	14)		3.2)

P = Pi =

(0.1312 0.1696 0.5569 0.0124 0.8283 0.5586
0.2329 0.4135 0.8307 0.3736 0.1004 0.9991
0.2348 0.1451 0.3522 0.2883 0.3047 0.6650
0.4047 0.8828 0.8732 0.5743 0.1091 0.0381)

The global minima is located at x = f(0.201690, 0.150011, 0.476874, 0.275332, ... 0.311652, 0.657301), $f(x*) \approx -3.32236$.

[8.] Quartic Function (Continuous, Differentiable, Separable, Scalable). $f(x) = \sum^{D}_{i=1} ix_i^4$ + random[0, 1) subject to $-1.28 \le xi \le 1.28$.

The global minima is located at x∗= f(0, · · · , 0), f(x∗) = 0.

[9] Hosc45: $f(x) = 2 - \prod_{10} xi / n!$ where x= (x1…10) and $0 \le xi \le i$ i =1with the $f(x^*) = 1$

[10.] Rosenbrock Function (Continuous, Differentiable, Non-Separable, Scalable, Mono-modal) $f(x) = \sum^{D-1}_{i=1} [100\ [(xi+1 - x2i\ )^2 +$

$(xi - 1)^2]]$ subject to $-30 \le xi \le 30$. The global minima is located at x∗= f(1, · · · , 1), f(x∗) = 0.

[11.] Shekel 5 (Continuous, Differentiable, Non-Separable, Scalable, Multimodal)

$f(x) = -\sum_{j=1}^{5} 1/\sum_{i=4}^{4}(xj - aij)^2 + ci$

where $A=A(A_{ij})$ =

4 4 4 4		0.1
1 1 1 1		0.2
8 8 8 8	C=C1 =	0.2
6 6 6 6		0.4
3 7 3 7		0.4

subject to $0 \le xj \le 10$. The global minima is located at x∗= f(4, 4, 4, 4), $f(x*) \approx -10.1499$.

[12.] Shekel 7 (Continuous, Differentiable, Non-Separable, Scalable, Multimodal)

$f(x) = -\sum_{j=1}^{7} 1/\sum_{i=4}^{4}(xj - aij) + ci$

4 4 4 4			0.1
1 1 1 1			0.2
8 8 8 8			0.2
6 6 6 6	,	c=c1=	0.4
3 7 3 7			0.4
2 9 2 9			0.6
5 5 3 3			0.3

subject to $0 \le xj \le 10$. The global minima is located at x∗= f(4, 4, 4, 4), $f(x*) \approx -10.3999$.

[13.] Shekel 10 (Continuous, Differentiable, Non-Separable, Scalable, Multimodal)
$f132(x) = -\sum_{j=1}^{10} 1/\sum_{i=4}^{4}(xj - aij)2 + ci$

where A = [aij] =

4 4 4 4		0.1
1 1 1 1		0.2
8 8 88		0.2
6 6 6 6		0.4
3 7 3	7 c=ci =	0.4
2 9 2 9		0.6
5 5 3 3		0.0
8 1 8 1		0.7
6 6 6 2		0.5
7 3.67 3.6,		0.5

subject to 0 ≤ xj≤ 10. The global minima is located at x∗= f(4, 4, 4, 4), f(x∗) ≈ −10.5319.

[14.] Shubert Function (Continuous, Differentiable, Separable, Non-Scalable, Multimodal)f(x) = ∏ ∑ cos((j + 1) xi+j) i=1 j=1, subject to −10 ≤ xi ≤ 10, i ∈1, 2,,n. The 18 global minima are located at
x∗= f({−7.0835, 4.8580}, {−7.0835,−7.7083},
{−1.4251,−7.0835}, { 5.4828, 4.8580},
{−1.4251,−0.8003}, { 4.8580, 5.4828},
{−7.7083,−7.0835}, {−7.0835,−1.4251},
{−7.7083,−0.8003}, {−7.7083, 5.4828},
{−0.8003,−7.7083}, {−0.8003,−1.4251},
{−0.8003, 4.8580}, {−1.4251, 5.4828},
{5.4828,−7.7083}, {4.8580,−7.0835},
{5.4828,−1.4251}, {4.8580,−0.8003}),
f(x∗) ≃ −186.7309.

[15.] F1 (1 variable):
f(x)=2(x−0.75)2+sin(5πx+0.4π)−0.125, where 0≤x≤1, with the $f(x^*)$ = -1.1232286

[16.] F3 (1 variable):
f(x)=−5i=1{i sin[(i+1)x+i]}, where−10≤x≤10 with the $f(x^*)$ = -12.0312494

[17.] F5n (20 variables):
f(x)=(π/20)×10 sin2(πy1)+[(yi−1)2×(1+10 sin2(πyi+1))]+(y20−1)2where x = [x1,...,x20]T,−10≤xi≤10 and yi=1+0.25(xi−1),
with the $f(x^*)$ = 0

[18.] F10n (20 variables):f(x)=π20×10sin2(πx1) +19i=1[(xi−1)2×(1+10sin2(πxi+1))]+(x20−1)2 where x =[x1,x2,...,x20]T and−10≤xi≤10 with the $f(x^*)$ = 0

[19.] F15n (20 variables):f(x) =(1/10)sin2(3πx1) +19i=1[(xi−1)2(1+sin2(3πxi+1))] + (1/10)(x20−)2[1+sin2(2πx20)] where x=[x1,x2,...,x20]T and −10≤xi≤10, with the $f(x^*)$ = 0

[19/20.] Pshubert 1 and 2: f (x,y) = {$\sum_{i=1}$((i cos i+1) x + i.) } x {$\sum_{i=1}^{5}(i\cos i+1)$y+ 1)} +β$(x+1:42513)^2$ $(y+1\ 0:80032)^2$
where -10 ≤ x ≤ 10 and -10≤ y≤ 10. B = 0.5 for Pshubert1, β =1 for Pshubert2

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
http://sciencedomain.org/review-history/9916