

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

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 discrete and continuous optimization problems. When compared to the Genetic Algorithm (GA) 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 [6], Artificial Bee Colony [3], Genetic Algorithm [2], Particle Swarm Optimization [1], 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 several months or even years. Some of these hitherto thought- to- be very difficult and time consuming problems include the travelling salesman's problem [3], job scheduling [2] and vehicle routing [2].

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 [3], inefficiency in the exploration of the search space[3], the use of several parameters[2], weakness in refining the search space at a later stage[1] and complex implementation strategies[4]. 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 kilometres 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-initialised. 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 and Improved Genetic Algorithm. The fourth section of the paper

presents the experimental results and 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 ('waaaa') behaviour of African buffalos in its foraging ventures. These are the two basic sounds of the African Buffalos with which they are able to organise themselves to search for food and defend themselves whenever they are attacked. The waaasound is used to mobilise the buffalos to move on to explore the search space while the maaa sound tells the buffalos to stay on to 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.

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 and also providing random speed. After this, the algorithm updates each buffalo's fitness and determines the bp_{max} and bg_{max} . 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

Figure 1. ABO algorithm is better than the herd's overall maximum, it saves it as the herd's maximum (bg_{max}). Finally it updates the

1. Objective function $f(x)$ $x=(x_1, x_2, \dots, x_n)_T$
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 k^{th} buffalo ($k=1,2,\dots,N$); lp_1 and lp_2 are learning factors; r_1 and r_2 are random numbers between $[0,1]$; 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.

buffalos speed and 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 in Figure 1.

A close look at the ABO algorithm in Figure 1, shows that the algorithm's movement equation has three parts, namely

the first w_k represents the memory of the buffalos past speed and 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 list of solutions of the buffalos memory instead of the present herd's maximum point. The second part $lp_1 r_1 (bg_{max} - m_k)$ 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 $lp_2 r_2 (bp_{max,k} - m_k)$ displays the intelligence of the animals and is a pointer to the buffalo's personal experience.

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 figure 1). This movement equation (1) provides for the actual movement of the herd using their specific speed. The speed-update (refer to equation (2)) provides for speed vector adjustment given 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 the first equation has two major parts, namely, the global maximum and the personal maximum positions: each defining the representative influence over the animal's speed. The algorithm subtracts the dimensional element w_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

123 fast convergence. Further research is on-going to get figures that may yield better results. The sum of these products is then
 124 added to the speed for the given dimension of the sector. This procedure is performed for each element of the speed vector. It
 125 should be emphasized that the random numbers give an amount of randomness in the path to help the animals move throughout
 126 the search space. It does this by randomly giving more or less emphasis to the global (*bgmax*)
 127

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.^[11]

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. 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 maximum 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 on two or more elements within the search space (or solution vector). The situation gets messier in a case of a multi-modal function. 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^[12].

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 stucked 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.

3.1 Chaotic Gray-coded Genetic Algorithm

In order to reduce the computational amount and improve computational precision for nonlinear optimizations, Xiaohua Yang (et al) developed the chaos gray-coded genetic algorithm (CGGA)^[16], 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^[15]. The main aim of this algorithm is to enhance the working of the standard Genetic resulting from the refinement of the evolution processes. In validating their algorithm, it was tested on a number of benchmark numerical functions.

3.2 Genetic Algorithm

Genetic Algorithm imitates the process of natural selection, evolution, inheritance, cross and mutation to arrive at solutions to optimization problems^[13]. 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 shall be investigated. The benchmark functions investigated in this study includes F1 function, F3, F5n, F10n, F15n, Branin, GoldPrice, Hosc45, Brown1, Camelback6, Shubert, Hartman1, Hartman2, PShubert1, PShubert2, Brown1, Shekel2, Shekel3, Brown3, Quartic and Rosenbrock^[12]. A brief description of each of the twenty-one benchmark test functions may be helpful in order to enable us appreciate them better.

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^[9b]. 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 bringing out the Relative Error percentage. A comparative lower relative error is a hallmark of good algorithm. This is calculated using the formula:

$$\text{Rel. Error} = \left| \text{Opt ABO} - \text{Opt R} \right| / \text{Opt R} * 100 \quad (5)$$

where Rel. Error is 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

$$\text{Rel. Error} = \left| \text{Opt ABO} - \text{Opt R} \right| * 100(6)$$

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. Finally the acceleration rate of the ABO is compared with the acceleration rate of other algorithms in order to evaluate the strength of ABO. This is obtained by the equation:

$$\text{Accel. Rate} = \text{FE}_{\text{algo}} / \text{FE}_{\text{ABO}}$$

where Accel. Rate refers to the acceleration rate, FE_{algo} is average function evaluation of the other algorithms, i.e. GA, IGA or CCGA and FE_{ABO} is the average functional evaluation of the African Buffalo Optimization algorithm. In this analysis, wherever Accel. Rate is greater than one, the ABO is faster and vice-versa.

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 African Buffalo Optimization (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 ^[15,16].

Table 1: ABO Experimental results comparison with GA and IGA

S/ N o	Func tion nam e	Op tim al Res ult	No of Var iabl es	Xte risti cs	ABO				GA				IGA				CCGA			
					Op t	Rel. Err or	A F E	S u cc -%	Op t	Rel Err or	A F E	S u cc -%	Op t	Re l Er ro r	A F E	S u cc -%	OP t	Rel Err or	A F E	S u cc -%
f1	Bran in	0.3 978	2	MN	0.3 978	0 %	1 5 2	1 0 0	0.3 978	0 %	81 25	8 1	0.3 978	0 %	20 40	1 0 0	0.3 978	0 %	30 0	1 0 0
f2	Bro wn1	2	20	UN	2.2 725	6. 8 %	8 0 3 9	8 7	43. 628 1	861 .2 %	68 44	0	8.5 516	32 7.5 %	12 86 4	0	1.9 987	0.0 65 %	31 9	1 0 0
f3	Bro wn3	0	20	UN	0.0 010	1 %	4 0 0 0	9 9	1.3 060	13 %	84 10	5	0.6 746	67. 5 %	10 68 57	5	0.0 695 2	69. 5%	70 12 9	8 2
f4	Cam elbac k6	- 1.0 316 3	2	MN	- 1.0 316	0 %	5 8 2	1 0 0	- 1.0 316	0 %	13 16	9 8	- 1.3 163	0 %	20 40	9 8	- 1.0 316	0 %	30 1	1 0 0
f5	Gold Price	3	2	MN	3.0 000	0 %	1 0 3	1 0 0	3.0 000	0 %	81 85	5 9	3.0 03	0.0 03 %	13 16	1 0 0	3.0 008	0.0 003 %	30 5	1 0 0
f6	Hart man	- 3.8	3	MN	- 3.8	0 %	9 1	1 0	- 3.8	0.0 001	19 93	9 4	- 3.8	0.0 4	16 80	1 0	- 3.8	0.1 2%	33 7	1 0

	3	627			627			0	625	%			611	%		0	677			0
f7	Hartman 6	- 3.3 223	6	MN	- 3.3 223	0 %	2 3 1	1 0 0	3.3 065	4.7 %	19 45 2	2 3	- 3.3 138	0.2 %	53 79 2	1 0 0	- 3.3 195	0.0 %	40 0 0	1 0 0
f8	Quartic	- 0.3 523	2	US	0.3 523	0 %	1 1 1	1 0 0	0.3 523	0 %	81 81 3	8 3	0.3 523	0 %	31 68 0	1 0 0	- 0.3 523	0 %	30 8 0	1 0 0
f9	Hoscar 45	1	10		2.0 000	10 0 %	1 4 5	5 0 5	1.9 951	99. 9 %	11 14 0	0	1.0 094	9 %	12 61 39	9 2	1.0 000	0 %	30 7 0	1 0 0
f10	Rosenbrock	0	2	UN	0.0 000	0 %	1 2 7	1 0 0	7.3 9	-	- - -	-	- - -	-	- - -	-	- - -	-	- - -	-
f11	Shekel5	- 10. 153 2	4	MN	- 10. 153 2	0 %	1 5 0	1 0 0	- 10. 134 9	1.7 %	74 95	1	- 10. 148 6	0.4 %	36 38 8	9 7	- 10. 152 0	0.0 %	60 0 0	1 0 0
f12	Shekel7	- 10. 402 9	4	MN	- 10. 402 9	0 %	2 2 8	1 0 0	- 10. 167 7	2.3 %	84 52	0	- 10. 382 5	1.2 %	36 77 4	0 8	- 10. 387 3	1.5 %	60 0 0	1 0 0
f13	Shekel10	- 10. 536 4	4	MN	- 10. 536 4	0 %	2 8 0	1 0 0	- 10. 034	0.1. 5 %	85 21	0	- 10. 514 0	0.2 %	36 77 2	1 0 0	- 10. 520 8	1.5 %	71 9 0	1 0 0
f14	Shubert	- 186 .73 09	2	MS	- 186 .73 09	0 %	3 3 0	1 0 0	- 186 .73 10	0.0 005 %	69 76	9 3	- 186 .72 80	0.0 %	23 64 0	1 0 0	- 186 .73 09	0 %	35 9 0	1 0 0
f15	F1	- 1.2 323	1	MS	- 1.2 323	0 %	4 4 0	1 0 0	- 1.2 323	0 %	55 66 0	1 0 0	- 1.2 323	0 %	78 4 0	1 0 0	- 1.2 323	0 %	30 0 0	1 0 0
f16	F3	- 12. 031 3	1	MS	- 12. 031 3	0 %	4 6 0	1 0 0	- 12. 031 2	0.0 009 %	53 47 0	1 0 0	- 12. 031 2	0 %	74 4 0	1 0 0	- 12. 312 3	2.3 %	30 0 0	1 0 0
f17	F5n	0	20	MS	0.0 140	1. 4 %	7 6 3	8 7 3	0.4 735	47. 3 %	87 25	0	0.0 001	0.0 %	99 94 5	1 0 0	0.0 000	0 %	86 7 0	1 0 0
f18	F10n	0	20	MS	0.0 000	0 %	7 2 0	1 0 0	7.8 352	783 .5 %	92 98	0	0.0 001	0.0 %	11 39 2	4 9	0.0 000	0 %	18 60 0	1 0 0
f19	F15n	0	20	MS	0.0 000	0 %	7 9 4	1 0 0	0.5 212	52. 1 %	95 41	0	0.0 003	0.0 %	10 24 13	1 0 0	0.0 000	0 %	85 9 0	1 0 0
f20	PShubert1	- 186 .73 09	2	MS	- 185 .04 9	0. 9 %	3 3	0	- 186 .73 00	4.8. %	71 92	6 3	- 186 .68 57	0.0 3 %	88 53	1 0 0	- 186 .73 09	0 %	50 9 0	1 0 0
f21	PShubert2	- 186 .73 09	2	MS	- 185 .22 6	0. 8 %	3 3	0	- 186 .73 10	0.5 4 %	73 03	5 9	- 186 .70 47	0.0 15 %	41 16	1 0 0	- 186 .73 09	0 %	42 9 0	1 0 0

In Table 1, No of Variables = number of variables required by each test function, Rel. Error = Relative Error, Opt = benchmark
Optimal Result obtained by the particular algorithm, AFE= Average number of Function Evaluation, Succ % =percentage
Success, UN = Uni-modal and Non Separable, UN= Uni-modal and Separable, MS= Multimodal and Separable, US=Unimodal
and Separable.

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(*f1,f4,f5,f6,f7,f8,f10-16,f18,f19*) and 99% accuracy in other 3(*f3,f20,f21*) test cases out of the 21 test functions, GA obtained the optimal result in 5(*f1,f4,f5,f8,f15*) of the test cases and very near optima in other 4(*f6,f13,f16,f21*) cases obtaining about 99%. The CCGA obtained the optimal result in 11 benchmark cases (*f1,f4,f8,f9,f14,f17-21*) and 99% accuracy in four other functions (*f2,f5,f7,f11*). In total, GA obtained the optimal result in 10 test functions. The IGA obtained above 99% in 16 (*f1,f4,f5,f6,f7,f11,f12,f13,f14,f15,f16,f17,f18,f19,f20,f21*) 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^[23b]. 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 *f3* where they obtained Relative Error of 67.7% and 69.5% respectively: that is a Uni-modal environment. Same applies to *f9*, whereas it best performance are in *f1,f4,f7,f15,f16*: 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, Unimodal Non-Separable to Unimodal 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 in all the algorithms obtained optimal result in *f15* and *f16* 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 *f1,f4,f5,f8,f14,f20* and *f21*. In the same vein, the posed a good performance in *f6* that uses three variables: *f11,f12*, and *f13* that use four variables and *f7* 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 *f2* and *f3* where the algorithms had their worst performances. The cases in *f17,f18* and *f19* are not much different. The only slight exception here is CCGA. 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^[20b], 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 (*f2,f4*). 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 utilises 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 CCGA 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.

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 unimodal, 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

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

References

- [1.]Angeline P. Evolutionary Optimization VersusParticle SwarmOptimization: Philosophy and Performance Differences. *Evolutionary programming* VII. 1998 Springer, Berlin, 601–610.
- [2] Bin Y, Zhong.Zhen Y, Baozhen Y (2009): An improved ant colony optimization for vehicle routing. *European Journal of Operational Research* 2009196, 171–176
- [3] Dorigo M.,TStützle*Ant Colony Optimization*. 1995 Cambridge: The MIT Press
- [4] Karaboga D., B Akay A Modified Artificial Bee Colony (ABC) Algorithm for Constrained Optimization Problems. *Apple Soft Computing* 2011 11(3), 3021–3031
- [5] Mezura-Montes E., J. Velázquez-Reyes , C.A. Coello A comparative study of differential evolution variants for global optimization. *Proceedings of the 8th annual conference on Genetic and evolutionary computation..* 2006 ACM Press, New York, pp 485–492
- [6] Reinelt G. TSPLIB - A Travelling Salesman Problem Library, *ORSA Journal on Computing*, 1991, 3(4) Fall, pp 376-384
- [7] Jshi Y., R. Eberhart. Parameter Selection in Particle Swarm Optimization.*Evolutionary programming* VII. 1998 Springer, Heidelberg, pp 591–600
- [8] Khaze S.R., I Maleki, S Hojjatkah, ABagheriniaEvaluation of the Efficiency of Artificial Bee Colony and the Firefly Algorithm in Solving the Continuous Optimization. *International Journal on Computational Science & Applications* 2013. (IJCSA) 3, 23-35 [9.] .Xin-She Y. Engineering Optimization: An Introduction with Metaheurisc Applications. 2010. New Jersey, USA: John Wiley and Sons Inc.
- [10] Xin'SheY. etal(Eds.). Swarm Intelligence and Bio-inspired Computation. 2013. Amsterdam: Elsevier Inc.
- [11.]Ali M.M, Khompatraporn C, Zambinsky C (2005):A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimization Test Problems
- [12.] Zambrano-Bigiarini (): Test Functions for Global Optimization. Available at http://www.rforge.net/doc/packages/hydroPSO/test_functions.html
- [13] Stochastic Global Optimization Algorithms Workshop, University of Canterbury, Christchurch, New Zealand, June, 2005
- [14]. J.M. Machado, S. Yang, S.L. Ho, and P. Ni, “A common Tabu search algorithm for the global optimization ofengineering problems,” *Comput. Methods Appl. Mech. Engr.*, vol. 190, pp. 3501–3510, 2001.
- [15]Andre, J. Dognon T. Siary P. (2001): An improvement of the standard genetic algorithm fighting premature convergence in continuous optimization
Advances in Software Engineering. (Elsevier Inc.) 32, pp41-60

[16] Yang X, Yang Z, Yin X. J. (2008): Chaos gray-coded genetic algorithm and its application for pollution source identifications in convection–diffusion equation. *Communications in Nonlinear Science and Numerical Simulation*. 13, 8, pp 1676–1688

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

Appendix 1

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

$$f(x) = (x_2 - 5.1x_1^2/4\pi^2 + 5x_1/\pi - 6)^2 + 10(1 - 1/8\pi)\cos(x_1) + 10$$

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

[2.]Brown1 (20 variables):

$$f(x) = \sum_{i \in J} (x_i - 3)^2 + \sum_{i \in J} (10^{-3}(x_i - 3))^2 - (x_i - x_{i+1}) + e20(x_i - x_{i+1}) \text{ where } J = \{1, 3, \dots, 19\} - 1 \leq x_i \leq 4 \text{ for } 1 \leq i \leq 20 \text{ and } x = [x_1, \dots, x_{20}]^T. \text{ The global minimum is located at } x^* = f(0, \dots, 0), f(x^*) = 2.$$

[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 $-1 \leq x_i \leq 4$. The global minimum is located at $x^* = f(0, \dots, 0)$, $f(x^*) = 0$.

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

349
350 [5.] Goldstein Price Function (Continuous, Differentiable, Non-separable, Non-
351 Scalable, Multimodal) $f(x) = [1 + (x_1 + x_2 + 1)2(19 - 14x_1 + 3x_2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)2(18 - 32x_1 +$
352 $12x_2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$
353 subject to $-2 \leq x_i \leq 2$. The global minimum is located at $x^* = f(0, -1)$, $f(x^*) = 3$.

354
355 [6.] Hartman 3 Function (Continuous, Differentiable, Non-Separable, Non-Scalable,
356 Multimodal) $f(x) = \sum_{i=1}^4 c_i \exp(-\sum_{j=1}^3 a_{ij}(x_j - p_i)^2)$
357 subject to $0 \leq x_j \leq 1$, $j \in \{1, 2, 3\}$ with constants a_{ij} , p_{ij} and c_i are given as

358
359 $A = A_{ij} = \begin{pmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix}$

363
364 $P = P_i = \begin{pmatrix} 0.3689 & 0.1170 & 0.2673 \\ 0.4699 & 0.4837 & 0.7470 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{pmatrix}$

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

369
370 [7.] Hartman 6 Function. (Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal)
371 $f(x) = \sum_{i=1}^4 c_i \exp(-\sum_{j=1}^6 a_{ij}(x_j - p_i)^2)$
372 subject to $0 \leq x_j \leq 1$, $j \in \{1, \dots, 6\}$ with constants a_{ij} , p_{ij} and c_i are given as

373
374 $\begin{pmatrix} 10 & 3 & 17 & 3.8 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix} \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix}$

378
379 $P = P_i = \begin{pmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5586 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{pmatrix}$

384
385 The global minima is located at $x^* = f(0.201690, 0.150011, 0.476874, 0.275332, \dots$
386 $0.311652, 0.657301)$, $f(x^*) \approx -3.32236$.

387
388
389 [8.] Quartic Function (Continuous, Differentiable, Separable, Scalable). $f(x) =$
390 $\sum_{i=1}^D x_i^4 + \text{random}[0, 1)$ subject to $-1.28 \leq x_i \leq 1.28$.
391 The global minima is located at $x^* = f(0, \dots, 0)$, $f(x^*) = 0$.

392
393 [9] Hosc45: $_{10}$
394 $f(x) = 2 - \prod_{i=1}^{10} x_i / n!$ where $x = (x_1 \dots x_{10})$ and $0 \leq x_i \leq 1$
395
396 with the $f(x^*) = 1$

397
398 [10.] RosenbrockFunction (Continuous, Differentiable, Non-Separable, Scalable,
399 Mono-modal) $f(x) = \sum_{i=1}^{D-1} [100 [(x_{i+1} - x_i^2)^2 +$
400 $(x_i - 1)^2]]$ subject to $-30 \leq x_i \leq 30$. The global minima is located at $x^* = f(1, \dots, 1)$, $f(x^*) = 0$.

402
403 [11.] Shekel 5 (Continuous, Differentiable, Non-Separable, Scalable, Multimodal)
404 $f(x) = -\sum_{j=1}^5 1 / \sum_{i=1}^4 (x_j - a_{ij})^2 + c_i$
405 where $A = A(A_{ij}) =$

406 $\begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \end{pmatrix} \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \end{pmatrix}$

410 $3 \ 7 \ 3 \ 7$ 0.4
411 subject to $0 \leq x_j \leq 10$. The global minima is located at $x^* = f(4, 4, 4, 4)$, $f(x^*) \approx$
412 -10.1499 .
413
414 [12.] Shekel 7 (Continuous, Differentiable, Non-Separable, Scalable, Multimodal)
415 $f(x) = -\sum_{j=1}^7 1 / \sum_{i=4}^4 (x_j - a_{ij}) + c_i$
416 $4 \ 4 \ 4 \ 4$ 0.1
417 $1 \ 1 \ 1 \ 1$ 0.2
418 $8 \ 8 \ 8 \ 8$ 0.2
419 $6 \ 6 \ 6 \ 6$, $c=c_1=$ 0.4
420 $3 \ 7 \ 3 \ 7$ 0.4
421 $2 \ 9 \ 2 \ 9$ 0.6
422 $5 \ 5 \ 3 \ 3$ 0.3
423 subject to $0 \leq x_j \leq 10$. The global minima is located at $x^* = f(4, 4, 4, 4)$, $f(x^*) \approx$
424 -10.3999 .
425
426 [13.] Shekel 10 (Continuous, Differentiable, Non-Separable, Scalable, Multimodal)
427 $f_{132}(x) = -\sum_{j=1}^{10} 1 / \sum_{i=4}^4 (x_j - a_{ij})^2 + c_i$
428 where $A = [a_{ij}] =$
429
430 $4 \ 4 \ 4 \ 4$ 0.1
431 $1 \ 1 \ 1 \ 1$ 0.2
432 $8 \ 8 \ 8 \ 8$ 0.2
433 $6 \ 6 \ 6 \ 6$ 0.4
434 $3 \ 7 \ 3 \ 7$, $c=c_i =$ 0.4
435 $2 \ 9 \ 2 \ 9$ 0.6
436 $5 \ 5 \ 3 \ 3$ 0.0
437 $8 \ 1 \ 8 \ 1$ 0.7
438 $6 \ 6 \ 6 \ 2$ 0.5
439 $7 \ 3.6 \ 7 \ 3.6$ 0.5
440 subject to $0 \leq x_j \leq 10$. The global minima is located at $x^* = f(4, 4, 4, 4)$, $f(x^*) \approx -10.5319$.
441
442 n_5
443 [14.] Shubert Function (Continuous, Differentiable, Separable, Non-Scalable, Multimodal) $f(x) = \prod_{i=1}^n \sum_{j=1}^n \cos((j+1)x_i + j)$ subject to
444 $-10 \leq x_i \leq 10$, $i \in 1, 2, \dots, n$. The 18 global minima are located at
445 $x^* = f(\{-7.0835, 4.8580\}, \{-7.0835, -7.7083\},$
446 $\{-1.4251, -7.0835\}, \{5.4828, 4.8580\},$
447 $\{-1.4251, -0.8003\}, \{4.8580, 5.4828\},$
448 $\{-7.7083, -7.0835\}, \{-7.0835, -1.4251\},$
449 $\{-7.7083, -0.8003\}, \{-7.7083, 5.4828\},$
450 $\{-0.8003, -7.7083\}, \{-0.8003, -1.4251\},$
451 $\{-0.8003, 4.8580\}, \{-1.4251, 5.4828\},$
452 $\{5.4828, -7.7083\}, \{4.8580, -7.0835\},$
453 $\{5.4828, -1.4251\}, \{4.8580, -0.8003\})$,
454 $f(x^*) \approx -186.7309$.
455
456 [15.] F1 (1 variable):
457 $f(x) = 2(x-0.75)^2 + \sin(5\pi x + 0.4\pi) - 0.125$
458 where $0 \leq x \leq 1$
459 with the $f(x^*) = -1.1232286$
460
461 [16.] F3 (1 variable):
462 $f(x) = -5i=1 \{i \sin[(i+1)x + i]\}$
463 where $-10 \leq x \leq 10$
464 with the $f(x^*) = -12.0312494$
465
466 [17.] F5n (20 variables):
467 $f(x) = (\pi/20) \times 10 \sin^2(\pi y_1) + [(y_i - 1)^2 \times (1$
468 $+ 10 \sin^2(\pi y_i + 1))] + (y_{20} - 1)^2$ where
469 $x = [x_1, \dots, x_{20}]^T$, $-10 \leq x_i \leq 10$ and $y_i = 1 + 0.25(x_i - 1)$
470 with the $f(x^*) = 0$
471

[18.] F10n (20 variables): $f(x) = \pi 20 \times 10 \sin 2(\pi x_1) + 19i = 1[(x_i - 1)^2 \times (1 + 10 \sin 2(\pi x_i + 1))] + (x_{20} - 1)^2$
 Where $x = [x_1, x_2, \dots, x_{20}]^T$ and $-10 \leq x_i \leq 10$
 with the $f(x^*) = 0$

[19.] F15n (20 variables): $f(x) = (1/10) \sin 2(3\pi x_1) + 19i = 1[(x_i - 1)^2 (1 + \sin 2(3\pi x_i + 1))]$
 $+ (1/10)(x_{20} - 1)^2 [1 + \sin 2(2\pi x_{20})]$ where $x = [x_1, x_2, \dots, x_{20}]^T$ and $-10 \leq x_i \leq 10$
 with the $f(x^*) = 0$

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

Final note

This article has been neither published nor submitted for publication, in whole or in part, either in a serial, professional journal or as a part in a book which is formally published and made available to the public