A Novel Intelligent Optimization Algorithm Inspired from Circular Water Waves

Muhammed Emre Colak[1], Asaf Varol[2]
[1]*Department of Software Engineering, Faculty of Engineering, Firat University, 23119 Elazig, Turkey*
[2]*Department of Software Engineering, Faculty of Technology, Firat University, 23119 Elazig, Turkey*
memrecolak@firat.edu.tr

***Abstract*—In this paper, a novel nature inspired algorithm for continuous optimization of numerical functions has been proposed. The algorithm is inspired from the circular wave created from a water drop when it falls to a still water. Algorithm itself has been developed with the purpose of distributed working thus kept as embarrassingly parallel as possible. Proposed algorithm is tested with eight different benchmark functions and up to thirty dimensions. The experimental results are promising and encouraging for further research studies.**

***Index Terms*—Optimization, nature-inspired computing, iterative methods, performance.**

I. INTRODUCTION

In today's scientific world; nature based algorithms have many benefits in optimization and demanding more research. Some of these algorithms are Particle Swarm Optimization (PSO) [1], Artificial Bee Colony (ABC) [2], Genetic Algorithms [3], Firefly Algorithm [4], Bat Algorithm [5], and Artificial Chemical Reaction Optimization Algorithm [6]. These algorithms are being improved by scientists and still open to more improvements. Some of the new research studies in this area are; Enhanced leader PSO [7], Enhanced Compact Artificial Bee Colony [8], the Intelligent Water Drops [9], and Modified Bat Algorithm [10].

By inspiring from nature, we may solve our problems. In this paper, a novel nature inspired algorithm for continuous optimization of numerical functions has been proposed.

In nature, when we observe a water drop to fall into still water, we will see circular waves forming around it. These waves will move until they collide with an object and this will create new waves. As waves travel through water by colliding with objects and creating new waves, we will see that waves are starting to gather near objects. As wave loses their velocity, they will vanish and leave water as still as ever.

We may assume that we have still water which can be interpreted as search domain and we have a mountain in it. We let water drop fall into it. This causes circular waves in the water. When a wave collides with a foothill, it will create new waves. These waves will cause new waves to form as it is colliding different parts of the foothill. As waves are moving in harmony, we should realize that forming waves are starting to gather near foothill. Now we have to make some assumptions for our algorithm sake.

– When wave collide with foothill, the water level will increase but formed waves will not be affected by it.

– When a new wave formed, it will not be a reflecting wave but a circular wave like its predecessors.

– Each wave has enough energy to travel as much as we want so they will not lose their energy as they move or reflect.

When the water level rises a newly formed wave will have chance to reflect from a higher place of the mountain. As this reflecting and level-rising continue; waves will make their way to the top of the mountain and will gather there. Our algorithm inspires from this water phenomenon.

This work has been organized as follows; at Section II the steps to optimize a given function with Circular Water Wave (CWW) algorithm are explained, at Section III benchmark functions, search domain of each function and best values of these function are provided, at Section IV experimental result of WCC, PSO and ABC are given and discussed, finally at Section V conclusion of the experimental result of Circular Water Wave algorithm and possible future research topics are given.

II. CIRCULAR WATER WAVE ALGORITHM

In this section, the steps to optimize a given function with Circular Water Wave algorithm (CWW) are explained. Constraints and parameters of the algorithm are given and discussed.

When we consider a function f with n variables and take the variables of function f as x_i ($0 \le i < n$), we will search maximum or minimum values of y, where

$$f\left(x_0, x_1, ...x_{n-2}, x_{n-1}\right) = y, \tag{1}$$

We will start with one point. Choose a random starting point S in search domain.

Since we cannot calculate every point in the search domain like circular water waves do pass, we have to make some sacrifices. Instead of calculating every point, we will calculate circular points in radius of r_i and we will do so in number of waves c_j ($1 \le j \le m$, m = maximum number of wave circle).

Manuscript received March 26, 2015; accepted August 2, 2015.

Calculate

$$d_i = r_i \times w \times c_j, \tag{2}$$

where d_i is the difference from previous value of x_i, r_i is the radius of i^{th} domain, w is a random value generator which is the type of double and has a value between 0 to 1, and c_j is the j^{th} circle. By using the difference d value, we shall create our circular search.

In order to make a smart search we shall calculate new function values of each dimension. For example; if we have three dimensional problem then we shall calculate and check fitness of:

$$\begin{cases} Point\ 1: (x_o + d_0, x_1, x_2), \\ Point\ 2: (x_o - d_0, x_1, x_2), \\ Point\ 3: (x_o, x_1 + d_1, x_2), \\ Point\ 4: (x_o, x_1 - d_1, x_2), \\ Point 5: (x_o, x_1, x_2 + d_2), \\ Point\ 6: (x_o, x_1, x_2 - d_2). \end{cases} \tag{3}$$

We create a new point with the points which has better fitness than starting point *S*. Let us assume Point 2 and Point 4 has better fitness than starting point *S* then we create new point as follows

$$Point\ 7: (x_o - d_0, x_1 - d_1, x_2). \tag{4}$$

Then we create additional points by randomizing this Point 7

$$Point\ 8: (x_0 - w \times d_0, x_1 - w \times d_1, x_2). \tag{5}$$

In our experiments, the number of this randomized possible best point is equal to the dimension of the problem.

r_i has an important factor in the results we obtained. We recommend the initial value of r_i to be equal of half of the range of the i^{th} dimension. When the fitness is not improving we update r_i values as follows

$$r_i = r_i / m. \tag{6}$$

The number of the wave circles is yet another important factor of the algorithm. The wave number and radius update function determine how fast algorithm approaches to solution. Increased wave numbers will minimize stacking at local minima. In our tests we have used value of wave circle as three, five and ten.

In an ideal state, we should use all new points which have better fitness than that of starting point to create new wave circles however, taking the best *b* value is recommended. In our tests, *b* value is selected as 3.

At each iteration, if the number of wave circle is *m*, dimension of the problem is *n*, and new starting point number is *b*; then the number of the points that each iteration need to calculate

$$m \times (3 \times n + 1) \times b. \tag{7}$$

Pseudo code of CWW algorithm:

Randomly choose starting points.
Do:
For each starting point
for each wave circle
-Calculate *di* values according to (2)
-Create new wave points in each direction and calculate fitness.
-Randomize best points
-Calculate fitness of new points.
If fitness is not improved
-Increment fail count and update r_i values according to (3)
Use new points which are best *b* to create new starting points.
While(fail count < 10)

III. BENCHMARK FUNCTIONS

Eight different benchmark functions [11] have been used in our experiments. Each of these functions has been tested for forty times. Functions are as follows:

F_1 Ackley's Function

$$f(x,y) = -20 \times \exp\left(-0.2 \times \sqrt{0,5 \times \left(x^2 + y^2\right)}\right) - \\ - \exp\left(0.5 \times \left(\cos\left(2\pi x\right) + \cos\left(2\pi y\right)\right)\right) + e + 20. \tag{8}$$

F_2 Beale's Function

$$f(x,y) = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + \\ + (2.625 - x + xy^3)^2. \tag{9}$$

F_3 Six Hump Camelback Function

$$f(x,y) = (4 - 2.1x^2 + \frac{x^4}{3}) \times x^2 + \\ + xy + (-4 + 4y^2) \times y^2. \tag{10}$$

F_4 Goldstein-Price Function

$$\begin{aligned} f(x,y) &= (1 + (x + y + 1)^2 \times \\ &\times (19 - 14x + 3x^2 - 14y + 6xy + 3y^2)) \times \\ &\times (30 + (2x - 3y)^2 \times \\ &\times (18 - 32x + 12x^2 + 48y - 36xy + 27y^2)). \end{aligned} \tag{11}$$

F_5 Lévi Function N.13

$$f(x,y) = \sin^2(3\pi x) + (x-1)^2(1 + \sin^2(3\pi y)) + \\ + (y-1)^2(1 + \sin^2(2\pi y)). \tag{12}$$

F_6 Eggholder Function

$$f(x,y) = -(y + 47) \times \sin\left(\sqrt{\left| y + \frac{x}{2} + 47 \right|}\right) - \\ - x \times \sin\left(\sqrt{\left| x - (y + 47) \right|}\right). \tag{13}$$

F_7 Hölder Table Function

$$f(x,y) = -\left|\sin(x)\cos(y)\exp\left(\left|1-\frac{\sqrt{x^2+y^2}}{\pi}\right|\right)\right|. \quad (14)$$

F_8 Sphere Function

$$f(x) = \sum_{i=0}^{n} x_i^2. \quad (15)$$

TABLE I. FUNCTIONS.

Function	Search domain	Minimum
F_1 (5)	-5 ≤ x, y ≤ 5	f(0, 0) = 0
F_2 (6)	-4.5 ≤ x, y ≤ 4.5	f(3, 0.5) = 0
F_3 (7)	-3 ≤ x_1 ≤ 3 -2 ≤ x_2 ≤ 2	f(0,0898, -0,7126) = 1,0316 f(-0,0898, 0,7126) = 1,0316
F_4 (8)	-2 ≤ x, y ≤ 2	f(0, -1) = 3
F_5 (9)	-10 ≤ x, y ≤ 10	f(1, 1) = 0
F_6 (10)	-512 ≤ x, y ≤ 512	f(512, 404.2319) = -959.6407
F_7 (11)	-10 ≤ x, y ≤ 10	f(±8.05502, ±9.66459) = -19.2085
F_8 (12)	−5.12 ≤ xi ≤ 5.12, i = 1, 2, …, n.	x* = (0, …, 0), f(x*) = 0.

IV. EXPERIMENTAL RESULTS

In this section, experimental results of WCC, PSO and ABC are given (in Table II–Table XI) and discussed.

Search domain and minimum value of the each function have been provided at Table I.

In Table II–Table IX, it is seen that the number of iteration is getting smaller as number of wave circle grows and as number of wave circle grows, mean values tend to get smaller which indicates less possibility of stacking local minima.

In general, algorithm manages to find absolute optimum point as seen in Table II–Table IX. In some cases as seen Table III–Table VII; algorithm may not reach absolute optimum but get close to it with an acceptable error rate.

TABLE II. EXPERIMENTAL RESULTS OF WCC FOR F_1 (5).

n	m	b	Mean Value	Best Value	Worst Value	Mean Iteration
2	3	3	2,042810365 31029E-15	0	3,552713678 8005E-15	53,95
2	5	3	9,769962616 70138E-16	0	3,552713678 8005E-15	49,35
2	10	3	1,154631945 61016E-15	0	7,105427357 601E-15	45,67

TABLE III. EXPERIMENTAL RESULTS OF WCC FOR F_2 (6).

n	m	b	Mean Value	Best Value	Worst Value	Mean Iteration
2	3	3	0,03549406 42440729	9,90556027 894557E-15	0,76206965 0892314	64,15
2	5	3	0,07047907 61674664	7,34711674 962036 E-20	1,17763273 524131	64,7
2	10	3	0,03873382 29740361	1,14488582 237879 E-13	1,15802752 960407	66,1

TABLE IV. EXPERIMENTAL RESULTS OF WCC FOR F_3 (7).

n	m	b	Mean Value	Best Value	Worst Value	Mean Iteration
2	3	3	1,03162845 348988	1,03162845 348988	1,03162845 348988	35,37
2	5	3	1,03162845 348988	1,03162845 348988	1,03162845 348988	33,42
2	10	3	1,03162845 348988	1,03162845 348988	1,03162845 348988	31,45

TABLE V. EXPERIMENTAL RESULTS OF WCC FOR F_4 (8).

n	m	b	Mean Value	Best Value	Worst Value	Mean Iteration
2	3	3	47,59269243 35569	3	87,5256658 408539	51,975
2	5	3	44,17081813 70669	3	89,0530442 264534	50,125
2	10	3	39,30261647 97705	3	86,6051568 007117	46,2

TABLE VI. EXPERIMENTAL RESULTS OF WCC FOR F_5 (9).

n	m	b	Mean Value	Best Value	Worst Value	Mean Iteration
2	3	3	1,349694649 63992E-31	1,349694649 63992E-31	1,349694649 63992E-31	57,5
2	5	3	1,349694649 63992E-31	1,349694649 63992E-31	1,349694649 63992E-31	52,05
2	10	3	1,349694649 63992E-31	1,349694649 63992E-31	1,349694649 63992E-31	46,975

TABLE VII. EXPERIMENTAL RESULTS OF WCC FOR F_6 (10).

n	M	b	Mean Value	Best Value	Worst Value	Mean Iteration
2	3	3	-915, 692830785117	-959, 640662720851	-633, 842302239904	40,375
2	5	3	-929, 747255522451	-959, 640662720851	-704, 805153877182	36,52
2	10	3	-947, 689995274587	-959, 640662720851	-888, 949125269878	30,3

TABLE VIII. EXPERIMENTAL RESULT OF F_7 (11).

n	m	b	Mean Value	Best Value	Worst Value	Mean Iteration
2	3	3	-19,2085 025678867	-19,2085 025678868	-19,2085 025678867	38,15
2	5	3	-19,2085 025678867	-19,2085 025678868	-19,2085 025678867	35,525
2	10	3	-19,2085 025678867	-19,2085 025678868	-19,2085 025678868	33,45

TABLE IX. EXPERIMENTAL RESULTS OF WCC FOR F_8 (12).

n	m	b	Mean Value	Best Value	Worst Value	Mean Iteration
2	3	3	0	0	0	470,3
2	5	3	0	0	0	426.875
2	10	3	0	0	0	368,1
10	3	3	0	0	0	548,2
10	5	3	0	0	0	548,2
10	10	3	0	0	0	548,075
30	3	3	1,26446320 988337 E-09	3,575242 2224758 E-129	2,84482586 613925 E-08	121.2
30	5	3	0	0	0	549,975
30	10	3	0	0	0	549,925

As seen in Table IX, as dimension of the problem grows; keeping wave circle number small may cause growing error rate. This could be avoided by incrementing wave circle number.

Since stopping criteria is the number of failed attempts, algorithm tends to iterating as long as it can find a new best solution which may cause increasing iteration number as seen in Table IX. However, this behaviour also gives algorithm a possibility to find absolute optimum point. In our experiment for F_8 regardless of dimension of the problem, we found variables with error rate of *e-162*.

Mean iteration numbers are promising and within acceptable range though algorithm may not be recommended as it stands for real time applications.

The number of the iterations tends not to change as the dimension of the problem grows but please take notice that as dimension grows so the number of the point to calculate in each wave circle as shown in (4).

In our tests, we have used randomized starting points within range of the search domain. One of the reasons for these randomized starting points is that some function has optimum minima at point zero and in order to test our algorithm in harsh environment, randomize starting points are chosen. In real world problems, using middle of the search domain as starting point is recommended. If more than one starting point are to be used; dividing search domain and putting starting points in middle of each part is recommended. Each starting point should move to possible best regardless of their sub search domain.

CWW algorithm may not be capable of finding the absolute minimum or maximum of every type of function. It may become necessary to try different radius values or to increase the number of the randomized best points. As seen in Table VI and IX, it may give result with some error or in worst case it may not be capable of solving it.

As it stands CWW algorithm is an intelligent algorithm that looks around of the given point and move through the best point which it can find.

In Table X, ABC algorithm (colony size: 20, max cycle: 200, and limit: 100) source code of which can be found in authors page [2] and in Table XI, PSO algorithm (particle number: 40, max cycle: 200, c_1: 1, and c_2: 1) have been used for testing some of the benchmark functions. According to these tables, results obtained from CWW algorithm are within the acceptable range.

TABLE X. RESULTS OBTAINED FROM ABC ALGORITHM.

Function	Dimension	Run Time	Mean Value	Best Value
F_1	2	40	3,998445e-017	0
F_2	2	40	5,105395e-004	2,648943e-006
F_6	2	40	-9.365490e+002	-9,596407e+002
F_7	2	40	-3,700000e+001	-3,797937e+001
F_8	10	40	1,417445e-014	2,484970e-017
F_8	30	40	8,314661e-004	8,09515e-009

TABLE XI. RESULTS OBTAINED FROM PSO ALGORITHM.

Function	Dimension	Run Time	Mean Value	Best Value
F_1	2	40	0,000817822997671236	0
F_2	2	40	0,0672485146195454	3,52620108288781E-06
F_3	2	40	-1,03146267393312	-1,03162822591744
F_5	2	40	0,00373558779877272	5,85130386838147E-06
F_6	2	40	-826,363040008659	-959,64066258274
F_8	2	40	2,81952458792166E-06	0

Our experiment reveals that CWW algorithm may be slower or faster than PSO and ABC, however usually has less error rate.

V. CONCLUSIONS

In this paper, a novel metaheuristic optimization algorithm entitled as CWW inspired from circular water waves, have been proposed. An intelligent approach to minimize points to calculate a circular search within a radius is used. Using best points founded by algorithm; new search points have been created and iterated until a better solution no longer can be found.

We have tested CWW algorithm with eight different benchmark functions and used some of functions to compare with ABC and PSO and seen that CWW has less error rate but also may need to perform more calculation.

CWW algorithm is a new born algorithm with much room for improvement. Some of the parts that demand improvement are; better mechanism for randomized best points, decision mechanism for radius, radius update mechanism, and decision mechanism for number of wave circles.

CWW algorithm has great capacity for finding absolute minimum or maximum values, which can be used to improve values which are found by other algorithms.

CWW algorithm can be used for hybridization with other algorithms to refine the search or minimize calculation.

CWW algorithm may be used for problems other than numerical functions with some modifications.

REFERENCES

[1] J. Kennedy, R. Eberhart, "Particle swarm optimization", in *Proc. IEEE Int. Conf. Neural Networks IV*, pp. 1942–1948, 1995.

[2] D. Karaboga, B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm", *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–171, 2007. [Online] Available: http://dx.doi.org/10.1007/s10898-007-9149-x

[3] D. E. Goldberg, *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley Publishing Company Inc.

[4] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*. Luniver Press, 2008, pp. 81–89.

[5] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*. Luniver Press, 2008, pp. 97–103.

[6] B. Alatas, "ACROA: artificial chemical reaction optimization algorithm for global optimization", *Expert Systems with Applications*, vol. 38, no. 10, pp. 13170–13180, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2011.04.126

[7] A. R. Jordehi, "Enhanced leader PSO (ELPSO): A new PSO variant for solving global optimisation problems", *Applied Soft Computing* vol. 26, pp. 401–417, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.asoc.2014.10.026

[8] A. Banitalebi, M. I. A. Aziz, A. Bahar, A. Aziz, "Enhanced compact artificial bee colony", *Information Sciences*. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2014.12.015

[9] H. Shah-Hosseini, "An approach to continuous optimization by the Intelligent Water Drops", *Procedia Social and Behavioural Sciences*, vol. 32, pp. 224–229, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.sbspro.2012.01.033

[10] S. Yilmaz, E. U. Kucuksille, Y. Cengiz, "Modified bat algorithm", *Elekronika IR Elektrotechnika*, vol. 20, no. 2, pp. 71–78, 2014. [Online] Available: http://dx.doi.org/10.5755/j01.eee.20.2.4762

[11] Virtual Library of Simulation Experiments: Test Functions and Datasets. [Online] Available: http://www.sfu.ca/~ssurjano/optimization.html