ISSN 2303-4521
Periodicals of Engineering and Natural Sciences *Original Research*
Vol. 11, No. 6, December 2023, pp.73-84

Hybrid cat swarm optimization with war strategy optimization for solving optimization problems

Sameerah Khaleel[1], Hegazy Zaher[2], Naglaa Ragaa Saeid[3]
[1] Department of Operations Research, Cairo University, Egypt
[2] Department of Statistics, Cairo University, Egypt
[3] Department of Operations Research, Cairo University, Egypt

ABSTRACT

The user-friendly and adaptable nature-inspired meta-heuristic algorithms have significantly contributed to their increasing appeal across several scientific fields, including computer science, mathematics, artificial intelligence, and operations research. The algorithms are constructed upon the core notions of exploration and exploitation. This study presents a fresh technique to optimizing issues by utilizing the War Strategy Optimization (WSO) method. The recommended technique involves utilizing the WSO algorithm in conjunction with the Cat Swarm Optimization (CSO) algorithm. The term used to describe this algorithm that combines different methods is C-WSO. The implementation of swarm intelligence approaches has led to the improvement of the capabilities of both algorithms.
A total of fifty benchmark test functions were adopted to assess the efficacy of the newly proposed C-WSO approach. Definite functions exhibited multimodality, while others were unimodal, and they were executed across diverse dimensions. Our investigations revealed that the C-WSO algorithm outperformed the original WSO approach. The method's performance has been assessed using a diverse range of measures, including the median, mean, and standard deviation of the fitness function values. Repeated evidence has exposed that the C-WSO approach surpasses the WSO algorithm in terms of effectiveness, creating it a viable and practical choice for solving complex optimization problems.

Keywords: Meta-heuristic Algorithms, War Strategy Optimization algorithm, Cat Swarm Optimization algorithm, Hybrid method

Corresponding Author:

Sameerah Khaleel
Department Of Operations Research
Faculty of Graduate Studies for Statistical Research, Cairo University,
Cairo, Egypt.
E-mail: samerakhaleel212@gmail.com

1. Introduction

To maximize or minimize an objective function while satisfying specific constraints, optimisation issues entail finding an optimal solution from a group of feasible alternatives. Math, CS, EE, OE, and operations research are just a few of the disciplines that have run across these issues. Linear, quadratic, and nonlinear optimisation problems fall within the category of continuous optimization, while the travelling salesman, knapsack, and graph-coloring issues are examples of discrete optimization. Algorithms and techniques like gradient descent, linear programming solvers, evolutionary algorithms, dynamic programming, and branch and bound approaches are commonly used to tackle optimization problems. The nature of the problem, including the variables, constraints, and objective function, dictates the optimization approach to be used. Efficiency, productivity, and efficacy can all be greatly enhanced through optimization [1].

Optimization algorithms that take their cues from the natural world are known as meta-heuristic algorithms. When faced with a problem space that does not have a clear solution, these algorithms will intelligently seek one out. They discover a happy medium between the time needed to find a good answer and the time needed to locate it. Over the course of several cycles or iterations, meta-heuristics employs predetermined rules or criteria to choose the optimal solution from a group of alternatives [2].

A group of computer methods known as swarm intelligence algorithms (SIA) took its cues from the coordinated actions of social insect colonies. Complex optimization and decision-making problems can be solved by using algorithms that emulate the decentralized and self-organized nature of these natural systems. They provide benefits including flexibility, resilience, and parallelism. Nevertheless, the issue at hand and the parameters that are fine-tuned determine the efficacy of a particular SIA [3]. One of such strategies is PSO, or particle swarm optimization [4],[5].The CSO (Cat Swarm Optimization) algorithm [6]. An algorithm known as the Ant Colony (AC) [7].A search for harmony (HS) [8]. Artificial Bee Colony (ABC) algorithm [9].This is known as the Bat Algorithm (BA) [10].The Flower Pollination Algorithm (FPA)[11],[12], and the Genetic Algorithm (GA) [13], [14] both of which rely on pollination and are according to Darwin's principle of natural selection (survival of the fittest). Algorithm for Cuckoo Search (CSA) [15].The Hunger Games Search (HGS) [16].Firefly Algorithm[17].

One characteristic shared by meta-heuristic approaches to optimization is the use of exploration and exploitation. When looking for answers in the search space, exploration entails venturing into uncharted territory, but exploitation is all about honing down on regions that are more likely to have solutions near the ideal ones [18].

2. War strategy optimization (WSO)

After researching historical military tactics, Ayyarao et al. (2022) [19] introduced WSO, a novel meta-heuristic optimization algorithm. Both offensive and defensive tactics are incorporated into this algorithm.

War strategy has always revolved around the king and commanders coordinating the movement of forces to accomplish their goals and objectives. As the armed forces work together and fight the enemy, war strategy is an ever-changing process. In response to evolving battlefield conditions, this tactic adjusts accordingly. The king's and commanders' positions have an ever-present impact on the troops' status. As all soldiers can see the flags flying from the royal and army commander's chariots, they can easily determine their whereabouts [20].The use of aural cues, such as drum beats or the cymbal crashes, is an integral part of soldier training. The strategy changes when a military leader dies, and the next commander has to figure out how to put the strategy back together while keeping it strong. The army's objective is to progress in rank by attacking the enemy side, while the king's objective is to defeat the enemy leader.

No matter how many times this happens, every soldier has an equal chance of becoming commander or king based on their fitness value. War is fought by both the King and the Commander. Following the King's and Commander's movements on the battlefield will help direct the remaining troops. The enemy's soldier, Local Optima, may be strong enough to ensnare the Leaders, posing a serious threat to either the King or the Commander. The position of the King or Commander, together with their combined movement methods, will direct the warriors in battle to avoid this.

2.1. Attack Strategy

It has come up with two plans for combat. Each soldier in the first plan revises his whereabouts according to where the King and Commander are. In Figure 1, we can see this attack model update method in action. In order to unleash a major attack on the adversary, the monarch takes use of an excellent position. Consequently, the highest-ranking soldier in terms of assault force or fitness is considered a monarch.

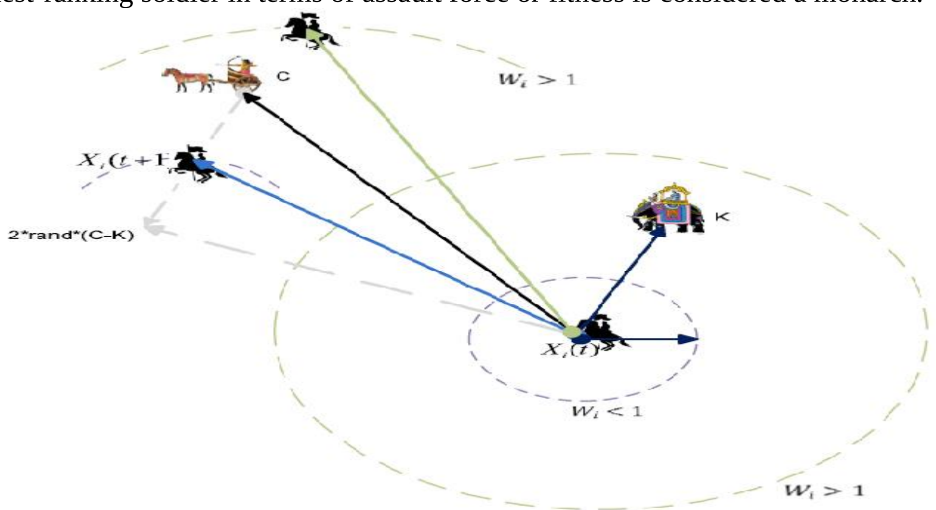

Figure 1. Attack Strategy [19]

In the beginning of the war, every soldier will be treated equally in terms of rank and weight. The soldier's rank is increased if the strategy is effectively carried out. All soldiers' ranks and weights will be adjusted depending on the strategy's performance as the fight advances. All of King, an army commander, and the soldiers are in adjacent proximity to the target as the fight draws to a close.

$$\mathcal{X}_{\mathrm{i}}(t+1) = \mathcal{X}_{\mathrm{i}}(t) + 2 \times \rho \times (C - K) + rand \times \left(W_{\mathrm{i}} \times K - \mathcal{X}_{\mathrm{i}}(t)\right) \ldots (1)$$

Where,

$\mathcal{X}_{\mathrm{i}}(t+1)$: is an updated position.

$\mathcal{X}_{\mathrm{i}}$: is the preceding C position for a commander.

K: is the position for the king.

$W_{\mathrm{i}} \times K - \mathcal{X}_{\mathrm{i}}(t)$ based on the king position. In the case of $W_{\mathrm{i}} > 1$, then a position for $W_{\mathrm{i}} \times K - \mathcal{X}_{\mathrm{i}}(t)$ is beyond the king position and hence the updated position of the soldier is beyond the commander position. If $W_{\mathrm{i}} < 1$, then a position for $W_{\mathrm{i}} \times K - \mathcal{X}_{\mathrm{i}}(t)$ is amid a king location and the soldier's current location. If W_{i} tends to zero, then a new position for a soldier transfers very close to a commander position, signifying the war's final stage.

2.2. Rank and weighing updating

The method by which the positions of the King, the Commander, and each soldier are updated determine the search agents' positions. W^k is impacted by the success record of every soldier on the battlefield, which is governed by equation (4). The ranking of a soldier (search agent) shows the relative proximity to the target (fitness value). If the assault force (fitness) in the soldier's new position (Fn) is lower than in their former position (Fp), the soldier will return to their original position.

$$\mathcal{X}_{\mathrm{i}}(t+1) = \left(\mathcal{X}_{\mathrm{i}}(t+1)\right) \times (F_{\mathrm{n}} \geq F_{\mathrm{p}}) + \left(\mathcal{X}_{\mathrm{i}}(t)\right) \times (F_{\mathrm{n}} < F_{\mathrm{p}}) \ldots (2)$$

In the case of a soldier updates a position successfully, a rank R_i for a soldier will be upgraded.

$$R_{\mathrm{i}} = (R_{\mathrm{i}} + 1) \times (F_{\mathrm{n}} \geq F_{\mathrm{p}}) + (R_{\mathrm{i}}) \times (F_{\mathrm{n}} < F_{\mathrm{p}}) \ldots. (3)$$

According to a rank, a new weight is considered as:

$$W_{\mathrm{i}} = W_{\mathrm{i}} \times \left(1 - \frac{R_{\mathrm{i}}}{Max\_iter}\right)^{\alpha} \ldots\ldots (4)$$

2.3. Defense strategy

According to the whereabouts of King, an army commander, and an unidentified soldier, the second strategy position update is calculated. However, there has been no change to the updating of rankings or weights.

$$\mathcal{X}_{\mathrm{i}}(t+1) = \mathcal{X}_{\mathrm{i}}(t) + 2 \times \rho \times \left(K - \mathcal{X}_{\mathrm{ran}}(t)\right) + rand \times W_{\mathrm{i}} \times \left(C - \mathcal{X}_{\mathrm{i}}(t)\right) \ldots (5)$$

2.4. Replacement or relocation of weak Soldiers

At the end of each cycle, the least fit soldiers are discarded and their positions filled by stronger ones. Swapping out the weak soldier for a random one is the easiest solution, as seen in the following equation.

$$\mathcal{X}w(t+1) = Lb + rand \times (Ub - Lb) \ldots (6)$$

The following is a description of how the WSO algorithm works:

Initialize the parameters as a first step.

Determine how many soldiers there will be.

Measure the problem's (the war space's) dimensions.

Limit the search space to a certain higher and lower limit.

Put the King (K) and the Army Commander (C) in their starting positions.

The King's and the Army Commander's attack forces must be defined.

Set the quantity of soldiers to one.

Second, set the variables to initial values:

Using the coordinates (1, soldier size), create a zero-vector R.

Make W a vector of length 1 (the size of a soldier) with 2 elements each element.

Third, in the battle area, disperse the troops uniformly and at random (random attack).

Fourth, using a for loop, iterate through the population of soldiers and get their assault forces.

Step5: Arrange the soldiers' attack forces according to their fitness.

The sixth step is to choose the most physically fit soldier to be king and the second fittest to be army commander.

Lastly, in Step 7, we initialize a counter variable t to 0.

Step8: Begin the main iteration loop as soon as t is less than the maximum iteration number. T max: Go through the entire population of soldiers iteratively: Make up a random integer ρ.

If the value of ρ is lower than a specified threshold ρr, then carry out the following exploration:

Put each soldier's current location into equation (5).

If that is not the case, then utilise the following:

Use equation (1) to update the positions of all soldiers.

Determine how many attackers each soldier will need.

Prioritize the health for each soldier.

Calculate an attack force for each soldier's current and past positions, and then update their positions using equation (2).

Make sure that each soldier's rank and weight are updated according to their success using equation (3).

A soldier iteration loop has come to a close.

Find the unfit soldier who is weakest.

Pick an appropriate relocation option and send the weak soldier somewhere else.

Please revise the King's and Commander's positions accordingly.

Add one to the counter t.

Step 9: The primary iteration loop comes to a close.

3. Cat swarm optimization (CSO) algorithm

The "seeking mode" and "tracking mode" that cats employ provide as inspiration for the CSO Algorithm. The position of each cat is comprised of M dimensions, with velocities for each dimension. Additionally, there is a fitness value that indicates the cat's fit with the fitness function, and a flag that indicates whether it is seeking or traversing mode. The optimal position in one of the cats would be the final solution because CSO keeps the best answer until iterations terminate [21].

3.1. Mathematical model of CSO

The scenario mimicked by the seeking mode is that of a cat, which involves resting, surveying its surroundings, and deciding where to go next. Searching memory pool (SMP), seeking range of the selected dimension (SRD), counts of dimension to change (CDC), and self-position considering (SPC) are the four fundamental components of seeking mode.

$$P_{\mathrm{i}} = \frac{|\, FS_{\mathrm{i}} - FSb|}{FS_{\max} - FS_{\min}}\,, \qquad where \qquad 0 < i < j \,....(7)$$

Finding the smallest possible answer is the objective of the fitness function, $FSb = FS_{\max}$ otherwise $FSb = FS_{\min}$. Tracing mode is a sub-model that mimics how a cat might act when following a predetermined trail of objects. The cat's every-dimension velocities determine its movement while in tracing mode [3].

$$v_{k,d} = v_{k,d} + r_1 \times c_1 \times (X_{best,d} - X_{k,d}), \text{ where } d = 1,2,..., M \text{ .... (8)}$$

$X_{best,d}$ is the position of the cat, who has the best fitness value; $X_{k,d}$ is the position of cat_k. c_1 is a constant and r_1 is a random value in the range of [0, 1].

$$X_{k,d} = X_{k,d} + v_{k,d} \quad \text{..... (9).}$$

The action of the CSO algorithm can be clarified by:

First thing to do: make N cats.

The second step is to randomly assign velocities to each cat within the range of the maximum velocity and to sprinkle them into the M-dimensional solution space. Then, select a few cats at random and put them into tracing mode when it comes to MR, while the rest go into seeking mode.

Step 3: Determine the fitness value of each cat by comparing their locations to the fitness function, which stands for the goal's criteria, and remembering the best cat. Remembering the location of the best cat (Xbest) is all that's needed because it represents the best solution up to this point.

Fourth, change the cats' modes of operation based on their flags; if the cat is searching mode, put it through its seeking mode procedure; otherwise, put it through its tracing mode process.

Step 5: Select the same amount of cats again and put them in tracing mode per MR's instructions; then, put the remaining cats in seeking mode.

Step 6: Enable the programme to exit, print the minimum and maximum flags, and verify that the termination condition is satisfied. In every other case, go back to steps 3–5.

4. Hybrid method

A hybrid method is one that optimises or solves a problem by combining numerous techniques or algorithms. The hybrid approach boosts the performance and overcomes the limits of swarm intelligence algorithms by integrating swarm intelligence techniques with other optimisation or search algorithms.
The hybrid method that results from combining swarm intelligence algorithms with additional techniques, such classic optimisation algorithms or search tactics, can offer numerous advantages. Better exploration-exploitation trade-offs, quicker convergence to optimal or near-optimal solutions, and enhanced solutions are all part of these advantages [22].The ACO-SA hybrid algorithm is one of many such hybrid algorithms [23].Combined ACO-DE [24].A hybrid method for mobile robot path planning [25].

5. Proposed new hybrid CSO with WSO (C-WSO)

An optimization algorithm so-called the CSO algorithm was formed by drawing influence from the natural hunting and foraging strategies of cats. It can efficiently explore a solution space and find optimal solutions to optimization problems by means of a population-based method.
The WSO optimization method is according to meta-heuristics to optimize optimization; it takes its cues from tactical force movements and classic war plans. A mix of global knowledge and local activity is necessary to find effective answers.
By joining CSO and WSO, we hope to make the most of the benefits that each method offers. The ability of CSO to perform global exploration—that is, to undertake extensive searches of the solution space—is a great complement to WSO's primary focus on local exploitation. The procedure for improving quality is described in the following way by Equation (6):

$$Xw\ (t+1) = min\ flags + rand \times (max\ flags - min\ flags) \ \dots \ (10).$$

This allows for the solutions to be modified and improved upon in various ways. Through the process of exploring and making use of the solution space, this combination is an effective instrument for addressing optimization issues because it searches for optimal solutions.

6. Computational experiments

A total of fifty benchmark test functions are utilized in order to evaluate the effectiveness of the proposed C-WSO. In this investigation, we provide both unimodal and multimodal functions, as shown in Tables 1 and 2, respectively.

Table 1. Details of unimodal benchmark functions

B.F.	Expression	F_{min}	Range	Dim.
F1	$f(x) = \sum_{i=1}^{n} x_i^2$	0	[-100,100]	50
F2	$f(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1]$	0	[-1.28,1.28]	50
F3	$f(x) = \sum_{i=1}^{D} \lvert Xi \rvert^{i+1}$	0	[- 1,1]	50
F4	$f(x) = \sum_{i=1}^{n} \lvert Xi \rvert$	0	[-100,100]	50
F5	$f(x) = max_i \sum_{i=1}^{n} \lvert Xi \rvert$, $1 \leq i \leq n$	0	[-100,100]	50
F7	$f(x) = 25 + \sum_{i=1}^{n} ([Xi])$	25-6n	[-5.12,5.12]	50

B.F.	Expression	F_{min}	Range	Dim.
F8	$f(x) = \sum_{i=1}^{n}(\sum_{j=1}^{i} x_j)^2$	0	[-100,100]	50
F9	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	0	[-100,100]	50
F10	$f(x) = \sum_{i=1}^{n} x_i^{10}$	0	[-10,10]	50
F11	$f(x) = \sum_{i=1}^{n-1}[(x_i - 1)^2 + 100(x_{i+1} - x_i^2)^2]$	0	[-30,30]	50
F12	$f(x) = \sum_{i=1}^{n-1}(x_i^2)^{(x^2_{i+1}+1)} + (x^2_{i+1})^{x_{i+1}}$	0	[-1,4]	50
F13	$f(x) = (x_1 - 1)^2 + \sum_{i=2}^{D} i(2x_i^2 - x_{i-1})^2$	0	[-10,10]	50
F14	$f(x) = \sum_{I=1}^{\frac{D}{4}} (x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4$	0	[-4,5]	50
F15	$f(x) = \sum_{i=1}^{n}(x_i)^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^4$	0	[-5,10]	50
F16	$f(x) = \exp\left(-\sum_{i=1}^{n}(x_i/\beta)^{2m}\right) - 2$	-1	[-20,20]	50
F17	$f(x) = \sum_{i=1}^{d}[\sum_{i=1}^{d}(j+\beta)(x_j^i - \frac{1}{j^i})]^2$	0	[$-d_i$, d_i]	5
F18	$f(x) = 2x_1^2 - 1.05x_1^4 + \frac{x_i^6}{6} + x_1x_2 + x_2^2$	0	[-5, 5]	2
F19	$f(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x^1 + x_1x_2^3)^2$	0	[-4.5,4.5]	2
F20	$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	0	[-10,10]	2
F21	$f(x) = (x_1 + 10)^2 + (x_2 + 10)^2 + e^{-x_1^2 - x_2^2}$	0	[-10,10]	2
F22	$f(x) = 0.26(x^{12} + x^{22}) - 0.48x_1x_2$	0	[-10,10]	2
F23	$f(x) = 0.5 + \frac{cos^2(\sin(\lvert x^2 - y^2 \rvert)) - 0.5}{(1 + 0.001(x^2 + y^2))^2}$	0.29257	[-100,100]	2
F24	$f(x) = 2\frac{x^{13}}{3} - 8x^{12} + 33x^1 - x^1x^2 + 5 + [(x^1 - 4)^2 + (x^2 - 5)^2 - 4]^2$	21.35	[-500,500]	2
F25	$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$	0	[-1.2,1.2]	2

Table 2. Details for multimodal benchmark functions

B.F.	Expression	F_{min}	Range	Dim.
F26	$f(x) = 418.9829n - \sum_{i=1}^{n} -x_i sin\sqrt{x_i}$	0	[-500,500]	50

B.F.	Expression	F_{min}	Range	Dim.
F27	$f(x) = \sum_{i=1}^{n} [x^2{}_i - 10\cos(2\pi x_i) + 10]$	0	[-5.12,5.12]	50
F28	$f(x) = 1 + \sum_{i=1}^{n} sin^2(x_i) - 0.1e^{(\sum_{i=1}^{n} x_i^2)}$	0.9	[-10,10]	50
F29	$f(x) = \sum_{i=1}^{n} (x^2 - i)^2$	0	[-500,500]	50
F30	$f(x) = \sum_{i=1}^{n} \lvert x_i \sin(x_i) + 0.1x_i \rvert$	0	[-10,10]	50
F31	$f(x) = \sum_{i=1}^{n} \in_i \lvert x_i \rvert^i$	0	[-5,5]	50
F32	$f(x) = -20\, e^{\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)} - e^{\left(\frac{1}{n\sum_{i=1}^{n}\cos(2\pi x_i)}\right)} + 20 + e$	0	[-32,32]	50
F33	$f(x) = \sum_{i=1}^{n} 8sin^2(7\{x_i - 0.9\}^2) + 6sin^2(14\{x_i - 0.9\}^2) + (x_i - 0.9)^2$	1	[-500,500]	50
F34	$f(x) = 1 - \cos\left(2\pi\sqrt{\sum_{i=1}^{n} x_i^2}\right) + 0.1\sqrt{\sum_{i=1}^{n} x_i^2}$	0	[-100,100]	50
F35	$f(x) = 1/2\sum_{i=1}^{n} (x_i^4 - 16x_i^2 + 5x_i)$	-39.16599	[-5,5]	50
F36	$f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{\sqrt{i}}}\right) + 1$	0	[-100,100]	50
F37	$f(x) = (\sum_{i=1}^{n} sin^2(x_i) - e^{\sum_{i=1}^{n} x_i^2})e^{\sum_{i=1}^{n} sin^2(\sqrt{x_i})}$	-1	[-10.10]	50
F38	$f(x) = (\sum_{i=1}^{n} \lvert x_i \rvert)e^{-\sum_{i=1}^{n} x_i^2}$	0	[-2π,2π]	50
F39	$f(x) = 0.1\left[sin^2(3\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2\{1 + sin^2(3\pi x_i + 1)\} + (x_n - 1)^2\{1 + sin^2(2\pi x_n)\}\right] + \sum_{i=1}^{n} u(x_i, 5,100,4)$	0	[-50,50]	50
F40	$f(x) = \pi/n\left[10sin(\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2\{1 + 10sin^2(\pi x_{i+1})\} + (x_n - 1)^2\right] + \sum_{i=1}^{n} u(x_i, 5,100,4)$	0	[-50,50]	50
F41	$f(x) = x^2 + y^2 + 25(sin^2(x) + sin^2(y))$	0	[-5.5]	2

B.F.	Expression	F_{min}	Range	Dim.
F42	$f(x) = -200e^{2\sqrt{x^2+y^2}} + 5e^{\cos(3x)+\sin(3y)}$	-195.629	[-32,32]	2
F43	$f(x) = \cos(x)\sin(y) - \frac{x}{y^2+1}$	-2.02181	[-1,2]	2
F44	$f(x) = \sin(x)\, e^{(1-\cos(y))^2} + \cos(y) e^{(1-\cos(x))^2}$	-106.7645	[-2π,2π]	2
F45	$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	-1.0316	[-5,5]	2
F46	$f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 - \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$	0.3978873	[-5,10]	2
F47	$f(x) = -\sum_{i=1}^{4} c_i e^{(\sum_{j=1}^{3} a_{ij}\{x_j - p_{ij}\}^2)}$	-3.862782	[0,1]	3
F48	$f(x) = \sum_{i=1}^{4} c_i e^{(\sum_{j=1}^{6} a_{ij}\{x_j - p_{ij}\}^2)}$	-3.32237	[0,1]	6
F49	$f(x) = -0.0001(\lvert\sin(x)\sin(y)\exp(100 - \frac{\sqrt{x^2+y^2}}{\pi})\rvert + 0.1)^{0.1}$	-2.062	[-10,10]	2
F50	$f(x) = \lvert x^2+\boldsymbol{y}^2+x\boldsymbol{y}\rvert+\lvert\sin(x)\rvert+\lvert\cos(\boldsymbol{y})\rvert$	1	[-500,500]	2

7. Results and discussion

To evaluate the effectiveness of C-WSO, we compare it to the WSO that was initially developed. In order to evaluate the performance of WSO and C-WSO, we do ten different runs of each benchmark function, each of which has a population size of fifty and a maximum of one thousand iterations. The mean, the standard deviation, and the median were the three performance evaluation metrics that utilised C-WSO. The mean was the best fitness value, and the standard deviation was the standard deviation. As can be seen in Figures 2-7, the findings indicate that C-WSO exceeds WSO in terms of performance.

Here is a quick rundown of how C-WSO and WSO compare:

F2, F4, F5, F6, F7, F9, F11, F13, F16, F17, F21, F23, and F24 are among the unimodal benchmark test functions where the suggested C-WSO outperforms WSO. All other functions yielded unaltered results.

F26, F29, F30, F31, F32, F34, F38, F39, F40, F44, F45, F46, F47, F48, F49, and F50 are among the multimodal benchmark test functions where the proposed C-WSO outperforms WSO. All other functions yielded identical findings.

The results showed that the suggested C-WSO improved the capabilities of exploration and exploitation operations, which means that meta-heuristics can accurately explore the search space. All of the measurements used in this study are compared between C-WSO and WSO in Figures 2–7. This optimization methodology can be adopted in all areas of science and technology including RF and microwave engineering [26-30].

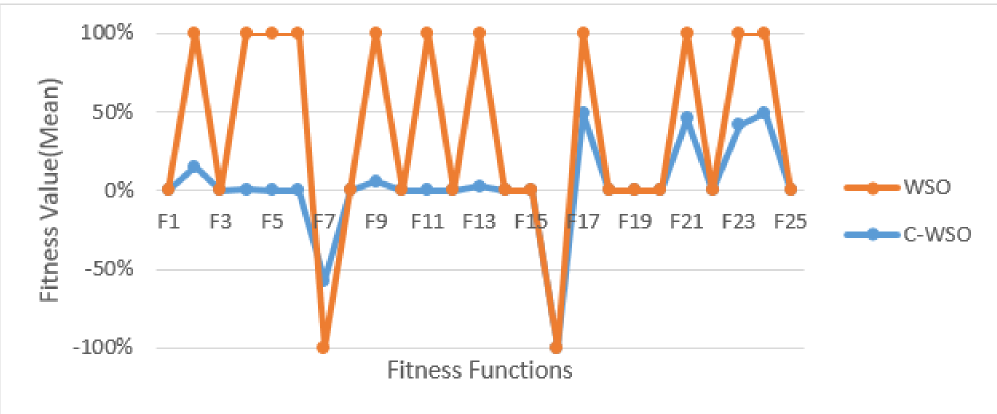

Figure 2.Comparison of Fitness Value (Mean) between WSO and C-WSO for unimodal functions

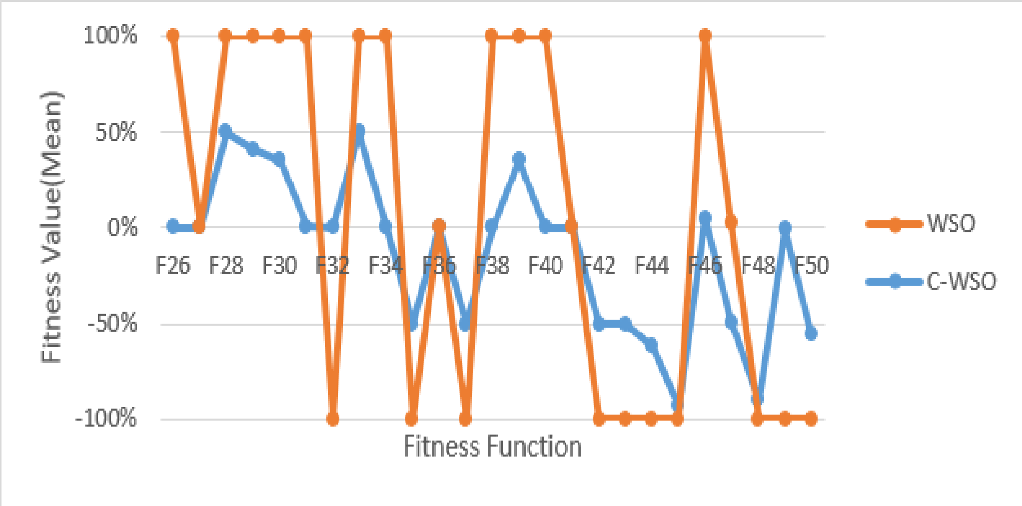

Figure 3. Comparison of Fitness Value (Mean) between WSO and C-WSO for multimodal functions

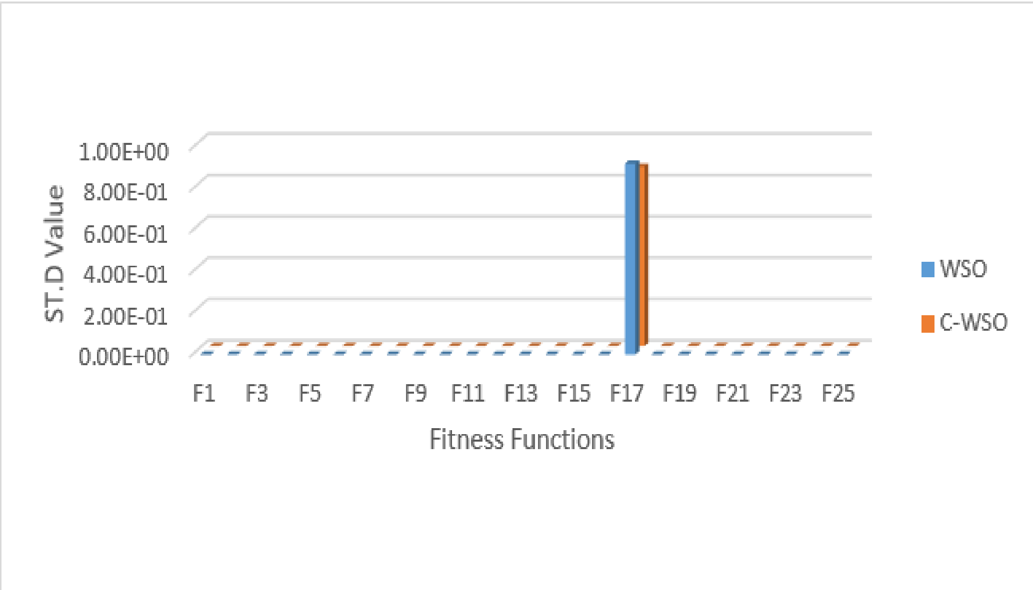

Figure 4. Comparison of ST.D. Value between WSO and C-WSO for unimodal functions

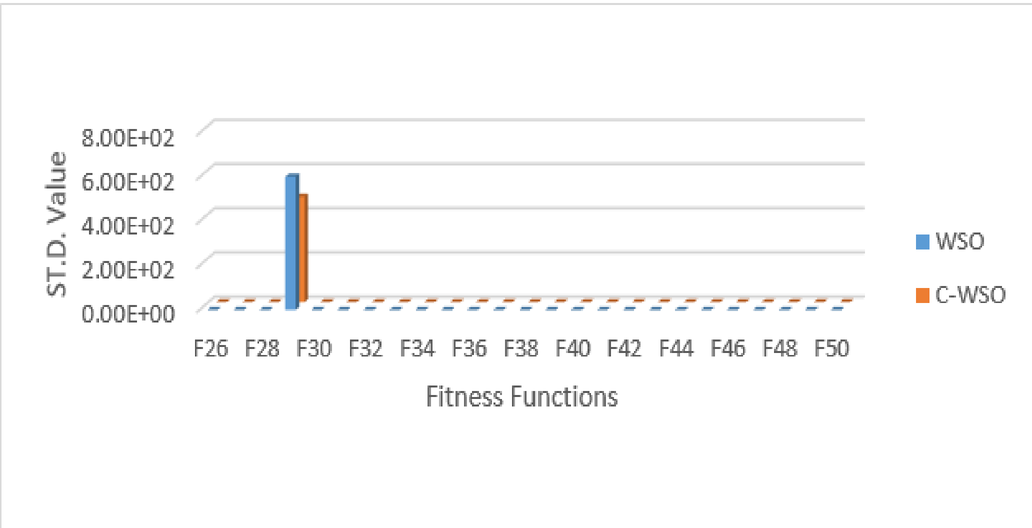

Figure 5. Comparison of ST.D. Value between WSO and C-WSO for multimodal functions

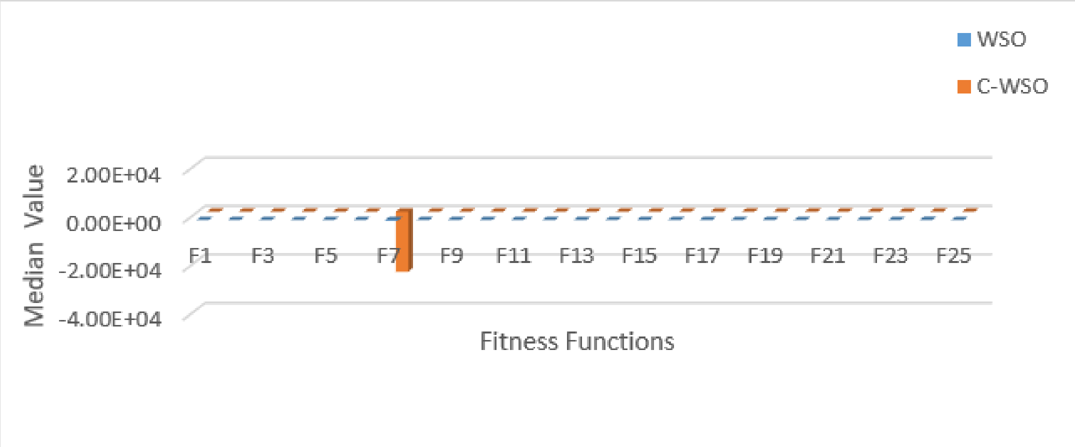

Figure 6. Comparison of Median Value between WSO and C-WSO for unimodal functions

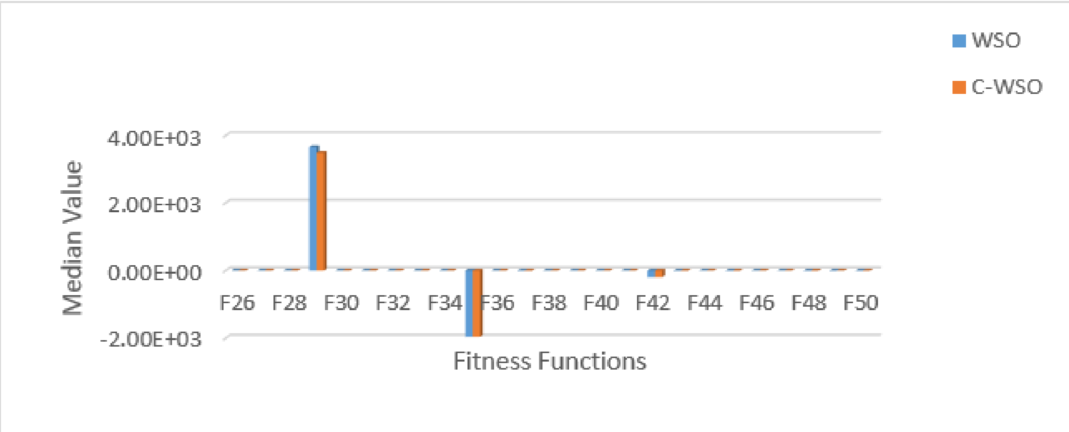

Figure 7. Comparison of Median Value between WSO and C-WSO for multimodal functions

8. Conclusion

Using a hybrid approach, the Water Search Optimization (WSO) algorithm has been made more efficient in solving optimization issues. A hybrid approach, C-WSO, combining WSO and CSO is suggested in this research. Fifty benchmark test functions were used to assess the performance of the suggested approach. When compared to the original WSO algorithm, the findings show that C-WSO is far better. Improved performance is evident from its higher fitness function values (Mean), St.d., and Median compared to WSO. This paper demonstrates that the hybrid C-WSO method outperforms WSO in terms of efficiency and performance by optimizing solutions for various test functions.

Declaration of competing interest

The authors declare that they have no known financial or non-financial competing interests in any material discussed in this paper.

Funding information

No funding was received from any financial organization to conduct this research.

References

[1] X. S. Yang, S. Deb, Y. X. Zhao, S. Fong, and X. He, “Swarm intelligence: past, present and future,” *Soft Comput.*, vol. 22, no. 18, pp. 5923–5933, 2018, doi: 10.1007/s00500-017-2810-5.

[2] D. M. El-sadek and others, “Solve global optimization problems based on metaheuristic algorithms,” *Bull. Fac. Sci. Zagazig Univ.*, vol. 2022, no. 3, pp. 29–42, 2022.

[3] Y. Ding, L. Chen, and K. Hao, *Bio-inspired optimization algorithms*, vol. 118. 2018. doi: 10.1007/978-981-10-6689-4_8.

[4] H. Garg, “A hybrid PSO-GA algorithm for constrained optimization problems,” *Appl. Math. Comput.*,

vol. 274, pp. 292–305, 2016.

[5] S. M. Redha, A. Abdullatif, and I. A. Hussain, “Using Particle Swarm Optimization Algorithm to Address the Multicollinearity Problem,” *ARPN J. Eng. Appl. Sci.*, vol. 14, no. 10, pp. 3345–3353, 2019, doi: 10.36478/JEASCI.2019.3345.3353.

[6] S.-C. Chu, P.-W. Tsai, and J.-S. Pan, “Cat swarm optimization,” in *PRICAI 2006: Trends in Artificial Intelligence: 9th Pacific Rim International Conference on Artificial Intelligence Guilin, China, August 7-11, 2006 Proceedings 9*, 2006, pp. 854–858.

[7] J. Handl, J. Knowles, and M. Dorigo, “Ant-based clustering and topographic mapping,” *Artif. Life*, vol. 12, no. 1, pp. 35–62, 2006.

[8] Z. W. Geem, J. H. Kim, and G. V. Loganathan, “A new heuristic optimization algorithm: harmony search,” *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[9] D. Karaboga and B. Basturk, “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm,” *J. Glob. Optim.*, vol. 39, pp. 459–471, 2007.

[10] T. K. Tharakeshwar, K. N. Seetharamu, and B. D. Prasad, “Multi-objective optimization using bat algorithm for shell and tube heat exchangers,” *Appl. Therm. Eng.*, vol. 110, pp. 1029–1038, 2017.

[11] X.-S. Yang, “Flower pollination algorithm for global optimization,” in *International conference on unconventional computing and natural computation*, 2012, pp. 240–249.

[12] O. Edin and S. Manfi, “Semi-parametric regression function estimation for environmental pollution with measurement error using artificial flower pollination algorithm,” vol. 13, no. 1, pp. 1375–1389, 2022.

[13] S. M. Redha and A. T. A. Hadia, “Employment of the genetic algorithm in some methods of estimating survival function with application,” *Period. Eng. Nat. Sci.*, vol. 8, no. 1, pp. 481–490, 2020, doi: 10.21533/pen.v8i1.1161.g530.

[14] D. E. Goldberg, “Cenetic Algorithms in Search,” *Optim. Mach. Learn.*, 1989.

[15] S. Feng, C. Cheng, and L. Mo, “An Improved Cuckoo Search Algorithm and Its Application in Function Optimization,” *Commun. Comput. Inf. Sci.*, vol. 1768 CCIS, no. 5, pp. 439–455, 2023, doi: 10.1007/978-981-99-0272-9_30.

[16] Y. Yang, H. Chen, A. A. Heidari, and A. H. Gandomi, “Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts,” *Expert Syst. Appl.*, vol. 177, p. 114864, 2021.

[17] S. Manfi, “Use the Firefly Algorithm to Find the Profits of the Airlines at Baghdad International Airport,” no. 2018, pp. 137–149.

[18] M. Dorigo, “Optimization, learning and natural algorithms,” *Ph. D. Thesis, Politec. di Milano*, 1992.

[19] T. S. L. V Ayyarao *et al.*, “War strategy optimization algorithm: a new effective metaheuristic algorithm for global optimization,” *IEEE Access*, vol. 10, pp. 25073–25105, 2022.

[20] U. N. I. V Er, S. Of, K. Library, and P. O. Rfof, “U N IV ER SITY OF NAIROBI KIKUYU LIBRARY P O RfOf 97 KIKUYU,” 2012.

[21] S. C. Chu and P. W. Tsai, “Computational intelligence based on the behavior of cats,” *Int. J. Innov. Comput. Inf. Control*, vol. 3, no. 1, pp. 163–173, 2007.

[22] T. O. Ting, X.-S. Yang, S. Cheng, and K. Huang, “Hybrid metaheuristic algorithms: past, present, and future,” *Recent Adv. swarm Intell. Evol. Comput.*, pp. 71–83, 2015.

[23] P. Stodola, K. Michenka, J. Nohel, and M. Rybansk\`y, “Hybrid algorithm based on ant colony optimization and simulated annealing applied to the dynamic traveling salesman problem,” *Entropy*, vol. 22, no. 8, p. 884, 2020.

[24] M. S. Iqbal, M. M. Kabir, A. S. Surja, and A. Rouf, “Solar Radiation Prediction using Ant Colony Optimization and Artificial Neural Network,” *Eur. J. Eng. Technol. Res.*, vol. 7, no. 2, pp. 99–111, 2022.

[25] T.-W. Zhang, G.-H. Xu, X.-S. Zhan, and T. Han, “A new hybrid algorithm for path planning of mobile robot,” *J. Supercomput.*, vol. 78, no. 3, pp. 4158–4181, 2022.

[26] J. K. Ali and Y. S. Miz’el, “A new miniature Peano fractal-based bandpass filter design with 2nd harmonic suppression,” in *2009 3rd IEEE International Symposium on Microwave, Antenna, Propagation and EMC Technologies for Wireless Communications*, 2009.

[27] Y. S. Mezaal and H. T. Eyyuboglu, “A new narrow band dual-mode microstrip slotted patch bandpass filter design based on fractal geometry,” in *2012 7th International Conference on Computing and Convergence Technology (ICCCT)*, IEEE, 2012, pp. 1180–1184.

[28] Y. S. Mezaal, H. H. Saleh, and H. Al-saedi, “New compact microstrip filters based on quasi fractal resonator,” *Adv. Electromagn.*, vol. 7, no. 4, pp. 93–102, 2018.

[29] F. Abayaje, S. A. Hashem, H. S. Obaid, Y. S. Mezaal, and S. K. Khaleel, “A miniaturization of the UWB monopole antenna for wireless baseband transmission,” *Periodicals of Engineering and Natural Sciences*, vol. 8, no. 1, pp. 256–262, 2020.

[30] Y. S. Mezaal and S. F. Abdulkareem, “New microstrip antenna based on quasi-fractal geometry for recent wireless systems,” in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, 2018.