Enhancing Optimization Efficiency: A Modified Sand Cat Swarm Optimization Algorithm with Adaptive Strategies and Dynamic Role Allocation

Prof. Vijay Kumar Bohat, Suman Lata, Avaya Aggarwal, Sreshth Khatri, Yuvraj Singh

Netaji Subhas University of Technology, Dwarka, Delhi, 110078, India

Abstract

Drawing inspiration from the sophisticated hunting tactics of sand cats, Sand Cat Swarm Optimization (SCSO) has gained prominence as an effective optimization method. Mirroring the sand cat's use of sound to locate prey, the algorithm employs similar strategies to identify optimal solutions. While SCSO demonstrates adeptness in pinpointing promising areas (exploitation), it occasionally encounters challenges in escaping local optima. To mitigate this limitation, we propose a Modified Sand Cat Swarm Optimization (MSCSO) algorithm. MSCSO incorporates an adaptive step size control strategy, which gradually reduces the step size over iterations, thus enhancing exploitation in later stages. Additionally, we introduce an enhanced selection process employing tournament selection to choose a set of k leaders . Moreover, we introduce a dynamic hunter-follower strategy, wherein a subset of the population alternates roles between hunters (focused on exploration) and followers (emphasizing exploitation) over time. Furthermore, we introduce a mechanism to identify potentially stuck solutions. By subjecting these solutions to a targeted random walk, leveraging the dynamic step size of SCSO, we enable exploration. This process utilizes either the best solution (for exploitation-oriented exploration) or selects another agent randomly (for diversification-oriented exploration). The optimization performances of the proposed SCSO algorithm are compared with well-known metaheuristic algorithms. The results show that the proposed algorithm can either outperform the compared metaheuristic-based algorithms or have competitive results.

Keywords: Sand Cat Swarm Optimization (SCSO), Modified Sand Cat Swarm Optimization (MSCSO), Optimization Algorithm, Metaheuristic Algorithm, Adaptive Step Size Control, Tournament Selection, Dynamic Hunter-Follower Strategy, Targeted Random Walk

1. Introduction

In real-world scenarios, numerous intricate problems persist, necessitating substantial computational efforts for efficient resolution [1, 2]. Optimization emerges as the preferred approach to tackle these challenges, aiming to attain optimal solutions. Metaheuristic methodologies stand

Email address: vijay.bohat@nsut.ac.in (Prof. Vijay Kumar Bohat, Suman Lata, Avaya Aggarwal, Sreshth Khatri, Yuvraj Singh)

out as widely acknowledged strategies for addressing such complexities. In essence, optimization entails maximizing efficacy and quality while minimizing time and cost. However, numerous intricate problems remain in need of optimization. Heuristic and metaheuristic approaches represent two distinct avenues for resolving these complexities and achieving optimal outcomes. While heuristic algorithms tailor solutions to specific problems, metaheuristic algorithms operate independently of specific problem structures, navigating predefined boundaries and random search spaces [3, 4]. Among these, Nondeterministic Polynomial Time (NP-hard) problems are prevalent. Metaheuristic algorithms, serving as potent optimization tools, find extensive applications across various domains [5]. These algorithms typically fall into several categories, including evolutionary, physics-based, human-behavior, and swarm intelligence algorithms [6].

Evolutionary algorithms, such as Genetic Algorithm (GA), constitute population-based search methods that leverage selection mechanisms to guide searches towards favorable solutions. Employing mutation and recombination, these algorithms operate on candidate solution strings to solve optimization or search problems [7, 8]. Additionally, Biogeography-Based Optimizer (BBO), Differential Evolution (DE), Tabu Search (TS), Black Widow Optimization (BWO), Gradient Evolution (GE), among others, offer diverse problem-solving approaches within this category [9, 10, 11, 12, 13]. Another category, based on physical principles, utilizes Physics-based algorithms. These algorithms adhere to physical laws, such as gravitational and electromagnetic forces, to navigate search spaces and identify optimal solutions. Examples include Gravitational Search Algorithm (GSA), Archimedes Optimization Algorithm (AOA), Lichtenberg Algorithm (LA), Henry Gas Solubility Optimization (HGSO), Fick's Law Algorithm (FLA), Kepler Optimization Algorithm (KOA), and Galaxy-based Search Algorithm (GbSA) [14, 15, 16, 17, 18, 19, 20].

Human-behavior algorithms, deriving inspiration from human behavior and social interactions, constitute another category of metaheuristic approaches. Algorithms such as Human Behavior-Based Optimization (HBBO), Human Mental Search (HMS), Mother Optimization Algorithm (MOA), Driving Training-Based Optimization (DTBO), Political Optimizer (PO), Teaching-Learning-Based Optimization (TLBO), Soccer League Competition (SLC), Social Mimic Optimization (SMO), and Battle Royale Optimization (BRO), reflect this category's diversity [21, 22, 23, 24, 25, 26, 27, 28, 29]. Lastly, Swarm Intelligence (SI) algorithms draw inspiration from collective behaviors observed in biological swarms. These algorithms mimic natural phenomena to solve complex optimization problems effectively. Notable examples include Particle Swarm Optimization (PSO), Fire Hawk Optimizer (FHO), Grey Wolf Optimization (GWO), Whale Optimization Algorithm (WOA), Bat Flower pollination (BFP), Ant Colony Optimization (ACO), Salp swarm Algorithm (SSA), Gazelle Optimization Algorithm (GOA), and Pelican Optimization Algorithm (POA) [30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40].

In addition to engineering, metaheuristic algorithms find application across diverse fields such as Medical image segmentation, feature selection, Machine Learning, Agriculture, and Software Engineering [41, 42, 43, 44, 45, 46]. Given the assertion of No Free Lunch (NFL) theorem, which posits that no single algorithm suffices for solving every optimization problem, there arises a need for new algorithms, modifications, and hybridization strategies tailored to specific problem contexts. Effective exploration-exploitation trade-offs and mitigating premature convergence pose ongoing challenges, requiring parameter tuning tailored to problem specifications.

In this paper, we propose an improved approach building upon the Sand Cat Swarm Optimization (SCSO) algorithm [47]. Despite SCSO's efficiency in real-life problem-solving, its slow convergence and susceptibility to local optima hinder its performance across various optimization problems. Our proposed method aims to address these limitations by enhancing the balance

between exploration and exploitation, thereby improving solution quality and convergence speed.

2. Original Sand Cat Swarm Optimization Algorithm

The Sand Cat Optimization Algorithm (SCOA) is a relatively new metaheuristic optimization algorithm inspired by the hunting behavior of sand cats in the desert. This algorithm was proposed by Aljarah et al. in 2022 [47].This algorithm shows the better result in terms of convergence rate in comparison to other swarm optimization algorithms.

2.1. Sand Cat in nature

The sand cat, a small wild cat found in deserts. The sand cat is similar to normal domestic cat in living behaviour except the structure of ear and paws.Unlike other animals, sand cat lives alone in dessert. the sand cat has remarkable ability to survive in harsh environments through efficient hunting strategies.The unique capability of sand cat is it's hearing power. It can hear the frequencies below 2 kHz which is incredible and helpful for hunting of the prey like small reptiles, small birds, mice, and insects etc.

2.2. Mathematical model

Every swarm intelligence algorithm is based on the two methods ,searching for the prey(exploration) and attacking the prey(exploitation).As sand cat lives alone in nature , for the mathematical modelling ,sand cats are assumed as herds to explain the concept of swarm intelligence.So for the algorithm initialization purpose, first declare the number of sand cats for the optimization.

2.2.1. Searching for the prey(exploration)

The beauty of the sand cat algorithm is it's dependency on fewer parameters for the optimization process.The algorithm is mainly dependent on the hearing factor of the sand cat.The SCSO algorithm procedure starts with the initialization of the population matrix, which is created according to the sand cat population and dimensions of the problem ($N_{\text{pop}} \times N_{\text{dim}}$), where N_{pop} is number of sand cats and N_{dim} is the size of the problem. The searching mechanism of the sand cat is based on the two parameters,the first parameter is hearing frequency 2 kHz, which is likely decreased to 0 after each iteration and the second parameter is a vector R, which is helpful for the successfully transition between exploration and exploitation phase.The general hearing sensitivity hearing range if described as

$$\vec{r}_{\text{G}} = S - (\frac{S \times iter_{\text{c}}}{iter_{\text{max}}}) \tag{1}$$

$$\vec{R} = 2 \times \vec{r}_{\text{G}} \times rand - \vec{r}_{\text{G}} \tag{2}$$

where S hearing sensitivity which is defined as 2, iter_{c} is the current iteration number,iter_{max} is the value of maximum iteration number and r is the random number between 0 and 1.

$$\vec{r} = \vec{r}_{\text{G}} \times rand \tag{3}$$

where is defined as the sensitivity range of sand cat which is different for every cat. Each sand cat updates its next position according to the best candidate position and its current position and it's sensitivity range .The exploration can be done according to the following equation:

$$P(\vec{t+1}) = \vec{r} \cdot (\vec{P_{\text{bc}}}(t) - rand \cdot \vec{P_{\text{cand}}}(t)) \tag{4}$$

where P_{bc} is best candidate position and P_{cand} is best position for that candidate.

2.3. Attacking on the prey(Exploitation)

The exploitation phase also depend on the hearing ability of the sand cat.This phase depends on the best global position and candidate's current random position.As we supposed the sensitivity range of a cat is in a circle , the movement direction is decided by an angle θ on the circle, which is chosen randomly and it's value range is between -1 and 1.This angle is decided by the theory of Roulettle Wheel selection algorithm.The randomness in this algorithm is used to avoid the problem of local optimal trapping.As the iteration progresses, the cat approaches to the prey to hunt. The exploitation can be done according to the following equation:

$$\vec{P(t+1)} = \vec{P_b(t)} - \vec{r} \cdot \vec{P_{rand}} \cdot \cos(\theta)) \tag{5}$$

where P_b shows the best global position of the sand cat and

$$\vec{P_{rand}} = |rand \cdot \vec{P_b(t)} - \vec{P_{cand}(t)}| \tag{6}$$

The exploration and exploitation transition depends on the value of vector R as accordingly.If $|R| \leq 1$, then expoltation will occur otherwise the exploration will be happened in the given search space.

2.4. Limitations

After thoroughly going through the sand cat algorithm the following drawbacks are found.

1. **Limited ability to escape local optima:** SCSO faces challenges in breaking free from local optima, which restrict its exploration of the solution space beyond local peaks. This limitation can prevent the algorithm from discovering globally optimal solutions, especially in complex optimization problems with rugged landscapes.
2. **Lack of adaptability in step size:** SCSO's fixed step size may not effectively accommodate the changing characteristics of the optimization landscape. As the algorithm progresses, a static step size can impede exploitation, as it might overshoot or undershoot promising regions. This lack of adaptability can diminish the algorithm's ability to converge towards optimal solutions efficiently.
3. **Inefficient selection process for leaders:** The process of selecting leaders in SCSO may not effectively identify the most promising solutions or individuals within the population. Inefficient leader selection can hinder the algorithm's ability to guide exploration and exploitation effectively, leading to suboptimal convergence rates or premature convergence to local optima.
4. **Limited balance between exploration and exploitation:** SCSO may struggle to strike an optimal balance between exploration (searching diverse regions of the solution space) and exploitation (focusing on promising areas). Without a proper balance, the algorithm may either get stuck in exploration without exploiting promising solutions or converge prematurely without fully exploring the solution space.
5. **Difficulty in identifying stuck solutions and applying appropriate exploration techniques:** SCSO may encounter challenges in detecting when solutions are stuck in local optima or stagnant regions of the search space. Without mechanisms to identify and address such situations, the algorithm may fail to apply appropriate exploration techniques, hindering its ability to discover better solutions.

6. **Low global search and population diversity:** SCSO's search strategy may lack the robustness needed to explore diverse regions of the solution space effectively. Insufficient population diversity can restrict the algorithm's ability to discover novel solutions and prevent it from thoroughly exploring the entire search space, potentially leading to suboptimal solutions or premature convergence.

3. Modified Sand Cat Swarm Optimizer

The goal of the modified Sand Cat Swarm Optimization (SCSO) algorithm is to make it better at finding the best solutions to problems. It does this in three main ways. First, it focuses more on the best-performing agents when choosing leaders, which helps pick the most capable leaders. Second, it balances between exploring new possibilities and exploiting known ones by adjusting how many agents are exploring versus exploiting at different stages of the process. Lastly, it helps the algorithm get out of situations where it's stuck in a less-than-ideal solution by trying out new options when it's not making progress. These changes make the algorithm more flexible and resilient, making it better at finding the best solutions.

3.1. Improvements

Enhanced Roulette Wheel Selection: The modified roulette wheel selection mechanism prioritizes the selection of the top k agents for leader designation. By placing greater emphasis on the best-performing sand cats, this adjustment aims to improve the quality of selected leaders and enhance the overall optimization process.

Dynamic Adjustment of Hunter Rate: In this adaptation, the proportion of hunters within the population gradually decreases over iterations. By reducing the number of hunters over time, the algorithm progressively shifts its focus towards exploitation, thereby potentially improving convergence and solution refinement in later stages of optimization.

Introduction of Targeted Random Walk for Stuck Solutions: To address challenges related to local optima, this modification introduces a targeted random walk mechanism (adjusted Levy Walk) for agents experiencing stagnation in fitness improvement. By incorporating controlled exploration when agents become trapped in suboptimal solutions, this strategy facilitates escape from local optima and promotes further exploration of the search space.

Table 1: Notation used in the algorithm

Symbol	Description
$X_(leader)$	Random leader position
X_r	Random position
X_t	Current position
X_{t+1}	Updated position
X_b	Best fitness position
T	Maximum iteration
t	Current iteration

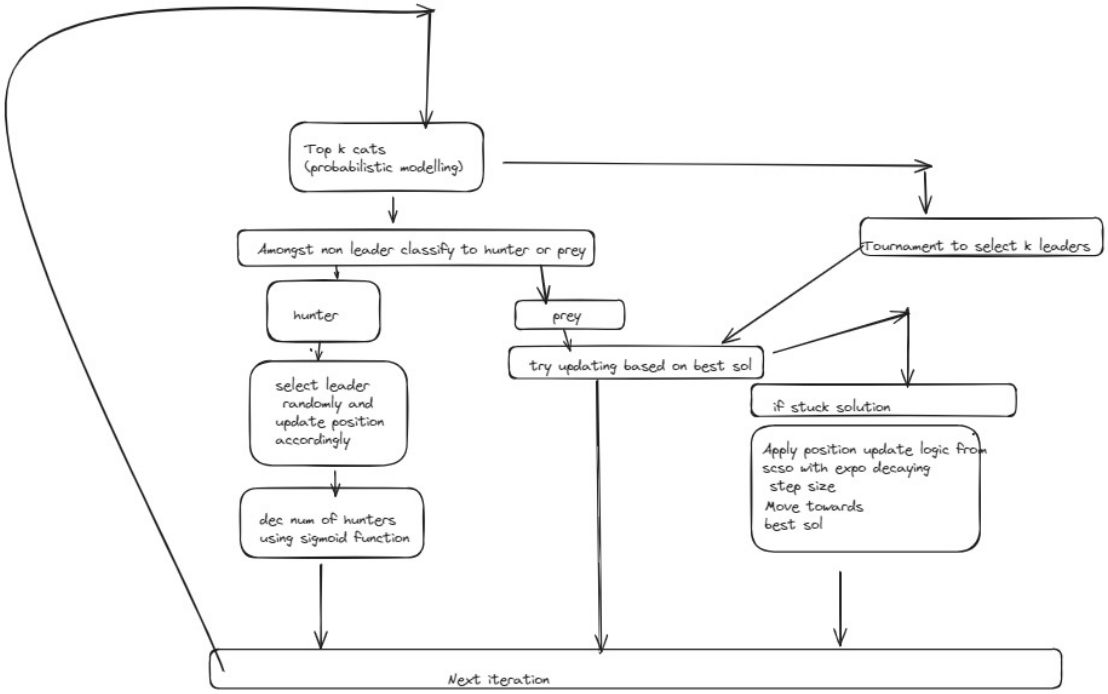


Figure 1: Flowchart of Modified SCSO

3.2. Selection (Modified Roulette Wheel)

The modified SCSO uses a modified roulette wheel selection focusing on the top k agents.The selection of the top k agents, also known as leaders,is accomplished through a tailored approach called modified roulette wheel selection. This method prioritizes agents based on their fitness, with higher probabilities assigned to those with superior fitness among the top k candidates. Essentially, the roulette wheel mechanism ensures that agents with better performance have a greater chance of being chosen as leaders, thus enhancing the quality of leadership within the algorithm.

3.3. Hunter Selection

The process of selecting hunters is governed by two key elements. First, a dynamic hunter˙rate is defined, which gradually decreases over iterations according to a sigmoid function. This dynamic adjustment ensures that the proportion of hunters within the population decreases over time. Second, a subset of agents is randomly selected to act as hunters, based on the calculated hunter˙rate. These hunters are tasked with exploration, probing new areas of the solution space. Meanwhile, the remaining agents become followers, focusing on exploiting known solutions to refine and improve them further. This dynamic selection process enables the algorithm to balance between exploration and exploitation effectively throughout the optimization process.

The algorithm also makes the use of Tournament Selection to implement a form of elitism by prioritizing agents with higher fitness values. It uses a randomized approach (selecting a random subset of agents to compete) to introduce some exploration while favoring better solutions.

sorted = sort(S, 'descend')
T = randperm(S˙sorted, k)
T :winner_index = argmax˙t $\in T\{S[t]\}$
leader_indices = $\{T[\text{winner\_index}_1], T[\text{winner\_index}_2], \ldots, T[\text{winner\_index}_k]\}$

The number of hunters is given as :

$$\text{midpoint} = 0.55 * T$$

$$\text{hunters} = \frac{\text{SearchAgents}}{(1 + e^{decay\ \text{rate}\cdot(t-\text{midpoint})})}$$

3.4. Position Updation

The modified SCSO incorporates an additional exploration mechanism using Levy flight as a random walk method. The targeted random walk for stuck solutions is a strategy to address situations where agents stagnate (minimal fitness improvement). This could involve techniques like moving closer to the best solution with a random offset and moving towards another randomly chosen agent's position.

Hunter behaviour: Firstly, a random leader is chosen from the selected leaders pool. Subsequently, the hunter moves away from this selected leader in a random direction. The distance of this movement is determined by a step size. By implementing these actions, hunters exhibit a proactive exploration strategy, as they venture away from the current best solutions in search of potentially superior alternatives within the solution space.

The step size is given by :

$$step = \alpha \cdot e^{-t \cdot \beta / T}$$

The position of hunters are updated as per :

$$\vec{X}_{t+1} = \vec{X}_t + \text{step} \cdot \text{sgn}(\vec{X}_t - \vec{X}_{\text{leader}}) \tag{7}$$

Follower behaviour: The agents update their positions based on the best solution with a combination of deterministic and stochastic factors. Initially, they adjust their positions according to the best solution with a predetermined step size, allowing for gradual convergence towards optimal solutions. Additionally, a random factor is introduced to inject variability into the movement, promoting exploration of the solution space.

To address situations where agents become stuck in suboptimal solutions, a targeted random walk mechanism is implemented. When the fitness change falls below a predefined threshold, indicating minimal improvement, agents employ one of two strategies. They either move closer to the best solution with a random offset, enhancing exploitation of promising regions in the solution space. Alternatively, agents may opt to move towards another agent's position, selected randomly.

The positions of non-hunters are updated as per:

$$\vec{X}_{t+1} = \vec{X}_t + (\vec{X}_b - \vec{X}_t) \cdot \text{step} \cdot \cos(\theta) \tag{8}$$

The condition checks used for non-hunters are:

$$|\text{fitness}(i) - \text{fitness}(i-1)| < \delta \tag{9}$$

$$R < 0.5 \qquad \vec{X}_{t+1} = \vec{X}_{\text{bc}} - \vec{X}_r \cdot \text{step} \cdot \cos(\theta)$$

$$R \geq 0.5 \qquad \vec{X}_{t+1} = \text{step} \cdot (\vec{X}_{\text{bc}} - \vec{X}_t \cdot \text{rand}(0, 1))$$

4. Literature Survey

5. Algorithms, test problems and comparison criteria

5.1. Algorithms

Grey Wolf Optimizer (GWO): Modeled after the hunting behavior of grey wolves, GWO simulates the hierarchical structure of wolf packs. It employs alpha, beta, and delta wolves to represent different roles within the pack, with alpha being the leader. By mimicking the collaborative hunting strategy of wolves, GWO navigates the search space effectively and converges towards the optimal solution. Its simplicity and demonstrated effectiveness make it a popular choice for benchmarking in optimization studies.

Whale Optimization Algorithm (WOA): Inspired by the cooperative feeding techniques of humpback whales, WOA mimics the bubble-net feeding and spiral foraging behaviors observed

in these marine mammals. By balancing exploitation and exploration, WOA optimizes the search process and efficiently explores promising regions of the solution space. Its innovative approach sheds light on how algorithms can effectively manage the trade-off between intensifying search efforts and diversifying exploration.

Gazelle Optimization Algorithm (GOA): Drawing from the collective movement patterns of mountain gazelles for safety, MGO emphasizes cooperation among individual solutions. By leveraging swarm intelligence principles, MGO encourages solutions to collaborate and converge towards the optimum collectively. This modern approach offers a contemporary perspective on nature-inspired optimization, showcasing the potential of collaborative strategies in problem-solving.

Political Sand Cat Optimization (PSCSO): Inspired by the territorial behavior of sand cats, PSCSO explores a less conventional strategy for optimization. By simulating the territorial dynamics observed in sand cat populations, PSCSO introduces unique insights into optimization techniques. This algorithm provides a novel perspective on leveraging animal behavior for computational problem-solving, offering potential benefits in specific optimization scenarios.

Crayfish Optimization Algorithm (COA): Modeled after the foraging behavior of crayfish, COA strikes a balance between resource acquisition and predator avoidance. Drawing inspiration from the innate survival instincts of crayfish, COA navigates the solution space efficiently while mitigating risks. Its distinct biological foundation offers a fresh perspective for comparison in optimization research, highlighting the importance of adaptability and risk management in algorithm design.

Gold Sine Algorithm based Sand Cat Algorithm (DGSSCSO): Representing a hybrid approach that combines the Gold Sine Algorithm with the Sand Cat Algorithm, DGSSCSO represents a cutting-edge advancement in Sand Cat Optimization. By integrating the strengths of both algorithms, DGSSCSO aims to enhance performance and efficiency in optimization tasks. This hybridization approach demonstrates the potential for synergistic combinations of nature-inspired algorithms to achieve superior optimization outcomes.

Each of these algorithms contributes a unique perspective to optimization research, showcasing the versatility and adaptability of nature-inspired approaches. Comparative analysis of their performance and characteristics offers valuable insights into the design and application of optimization techniques across diverse problem domains.

5.2. Test Problems

The IEEE Congress on Evolutionary Computation (CEC) is a recognized benchmark for evaluating the performance of optimization algorithms. The CEC 2014 and CEC 2017 competitions specifically offered test suites comprised of various benchmark functions. These functions encompass a diverse range of complexities, including unimodal, multimodal, hybrid, and composite problems. Utilizing these CEC functions as test criteria allows researchers to objectively assess the strengths and weaknesses of their proposed algorithms. By analyzing performance on functions with varying characteristics, researchers gain valuable insights into the algorithm's ability to handle different optimization landscapes. The CEC benchmark functions, particularly those from CEC 2014 and CEC 2017, serve as a cornerstone for establishing a common ground for algorithm comparison and advancement within the field of optimization.

CEC 2014 focused on introducing a new suite of 30 benchmark functions, categorized into unimodal, multimodal, composite, and hybrid problems. Unimodal functions have a single

global optimum, allowing assessment of an algorithm's convergence speed and accuracy. Multimodal functions, on the other hand, contain multiple local optima, challenging the algorithm's ability to escape from suboptimal solutions and locate the true global minimum. Composite functions combine elements of both unimodal and multimodal problems, offering a more realistic representation of many real-world optimization tasks. Finally, hybrid functions introduce additional complexities such as rotated or shifted versions of existing functions, further stressing the robustness of the algorithms.

CEC 2017 built upon the foundation laid by CEC 2014, retaining the core function categories while introducing variations and complexities. This continuity allows for comparisons between algorithms tested on both sets, but the variations in CEC 2017 functions provide new challenges. For instance, the introduction of functions with different separability characteristics (how easily the function can be decomposed into simpler sub-functions) can reveal how well algorithms handle problems with varying degrees of interdependencies between variables.

In essence, the CEC 2014 and CEC 2017 benchmark functions, with their diverse characteristics, offer a comprehensive suite for evaluating the capabilities of optimization algorithms. By analyzing an algorithm's performance on these functions, researchers gain valuable insights into its strengths and weaknesses in terms of convergence speed, ability to handle multiple optima, and robustness to complexities. This ultimately paves the way for the development of more efficient and effective optimization algorithms for tackling real-world problems.

5.3. Test Criteria

In the realm of optimization research, the comparison of algorithms is a fundamental aspect aimed at identifying superior performers for various problem domains. To ensure robust and reliable evaluations, it is essential to employ comprehensive comparison criteria that encompass both quantitative measures and statistical analyses. Two commonly utilized criteria in algorithm comparison are the average fitness obtained over multiple runs and statistical tests such as the Wilcoxon test with associated p-values.

Firstly, the average fitness obtained over multiple runs serves as a primary quantitative measure for assessing the performance of optimization algorithms. Conducting multiple runs of each algorithm on the same set of benchmark functions enables researchers to account for the stochastic nature of optimization processes. By computing the average fitness value across these multiple runs, we obtain a more representative indication of the algorithm's performance, reducing the influence of random fluctuations and outliers. This criterion allows for a robust comparison of algorithms in terms of their convergence behavior, solution quality, and consistency across different problem instances.

Secondly, statistical tests such as the Wilcoxon test play a crucial role in determining the significance of observed differences in algorithm performance. The Wilcoxon test, a non-parametric statistical test, is particularly well-suited for comparing the performance of two algorithms across multiple problem instances or benchmark functions. By analyzing the ranked differences in performance metrics (e.g., average fitness) obtained from repeated experiments, the Wilcoxon test assesses whether the observed differences are statistically significant or merely due to chance. The resulting p-value indicates the probability of obtaining the observed differences under the null hypothesis of no true difference between the algorithms. A low p-value (¡ 0.05) suggests

strong evidence against the null hypothesis, indicating that the observed differences in performance are unlikely to occur by random chance alone. Conversely, a high p-value (¿ 0.05) indicates that the observed differences are not statistically significant, thereby failing to reject the null hypothesis.

6. Experimental results and discussion

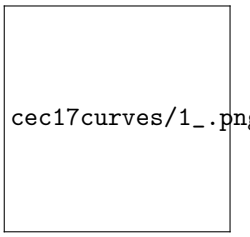
6.1. Performance Comparison of SCSO and MSCSO on CEC2014 and CEC2017

	CEC2017 (MEAN RESULTS ON 50 RUNS)			CEC2014 (MEAN RESULTS ON 50 RUNS)		
	SCSO	MSCSO	P VALUE	SCSO	MSCSO	P VALUE
F1	4231025.60608641	4617.18033780755	7.07E-18	300880.2763	66023.80933	1.60E-
F2	NA	NA	NA	3994809.729	6061.320114	7.07E-
F3	624.444214205	467.756982470505	5.64E-17	738.393004	473.6521456	2.53E-
F4	414.684632788782	414.070111912333	1.21E-08	408.2984864	403.8071893	1.88E-0
F5	533.018593878848	514.723239228615	0.11205464	519.9931814	519.6946496	1.35E-
F6	610.392156210242	600.012636216632	4.01E-09	601.6389591	603.0187046	2.13E-0
F7	756.519117468808	726.706546965491	0.319173882	700.9507355	700.5820825	1.49E-0
F8	828.514627111496	816.384584806365	0.1586028331	806.6789925	807.41803	0.81200768
F9	980.428242180097	901.345672239184	6.34E-17	914.0825507	916.5901018	0.030150595
F10	1937.78988224019	1500.15961896767	0.9697546486	1098.042925	1170.50471	0.004757371
F11	1161.63875738028	1120.33310182513	0.4628318402	1594.219347	1622.274104	0.3225344
F12	934902.913137379	559346.781000977	1.23E-09	1200.567674	1200.294219	9.46E-
F13	13642.3880943800	12149.0906274433	5.92E-11	1300.228171	1300.20402	0.10301943
F14	1990.44389940792	3307.36719654484	2.51E-14	1400.17366	1400.154276	0.018907704
F15	2546.47856383439	4271.08450088348	4.01E-16	1502.375395	1501.937685	0.000505297853
F16	1744.20684539039	1706.01051606479	0.9258506767	1602.760749	1602.877639	0.060303788
F17	1761.02125879070	1744.32168807032	0.001269172183	2571.771204	2262.940877	1.84E-
F18	17199.5049496006	14877.1295130167	1.19E-14	1889.017194	1877.710526	0.038303425
F19	4242.83580087805	7308.89778977917	6.20E-11	1901.999317	1902.012261	0.893062965
F20	2102.36435707472	2016.25056326115	0.2083475921	2054.044006	2031.389122	3.02E-
F21	2259.69165838203	2285.28993747737	3.06E-18	2434.357949	2296.736098	2.21E-0
F22	2305.04142050367	2294.85178788600	NaN	2223.281773	2221.306415	2.99E-0
F23	2633.64890069197	2620.94457338122	0.1250604265	2577.0601	2549.164879	2.99E-0
F24	2707.12078684670	2723.24520950964	2.68E-12	2520.556358	2522.359708	0.082954339
F25	2929.97817835856	2934.91952143191	0.696905702	2641.767658	2636.949789	0.006956270
F26	3006.75135259927	2983.03934775027	4.31E-10	2700.224484	2700.18401	0.000143566536
F27	3098.90286105166	3097.06461838267	0.4218995906	2704.418578	2706.411162	1.34E-0
F28	3286.43027356842	3263.03286170852	0.0002690771513	3177.768275	3175.839571	0.160648
F29	3221.42166026456	3212.29440739210	0.242604144	3310.464308	3215.387373	1.48E-
F30	533158.697421350	325338.055741814	2.16E-13	3630.203666	3595.495106	0.048259138

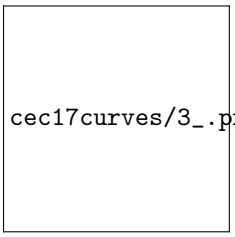
Function	**MSCSO vs SCSO** p-value	**MSCSO vs SCSO** h-value	**MSCSO vs PSCSO** p-value	**MSCSO vs PSCSO** h-value	**MSCS**
Row 2, Column 1	Row 2, Column 2	Row 2, Column 3	Row 2, Column 4	Row 2, Column 5	Row
Row 2, Column 1	Row 2, Column 2	Row 2, Column 3	Row 2, Column 4	Row 2, Column 5	Row
Row 2, Column 1	Row 2, Column 2	Row 2, Column 3	Row 2, Column 4	Row 2, Column 5	Row
Row 2, Column 1	Row 2, Column 2	Row 2, Column 3	Row 2, Column 4	Row 2, Column 5	Row

6.1.1. Convergence Curves for CEC2017

curves˙2017/ image2 image3
image4 image5 image6
image7 image8 image9
image10 image11 image12

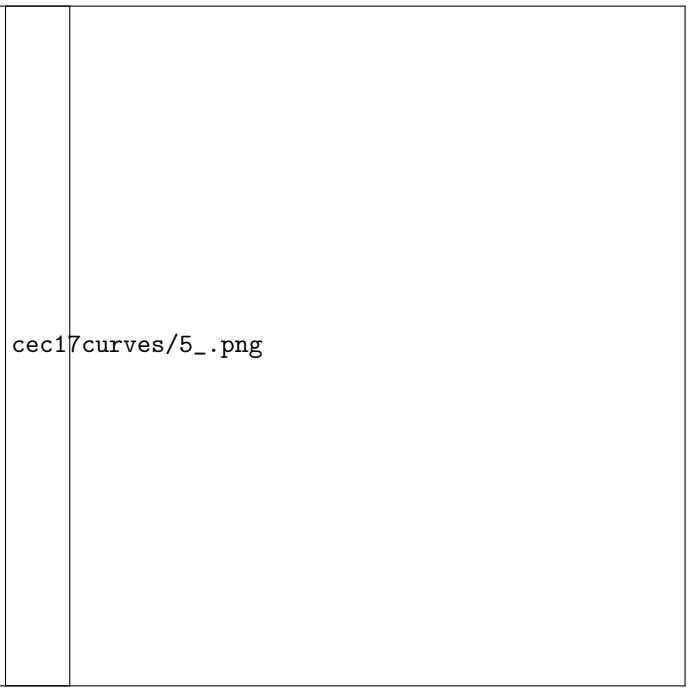


CEC2017-F1



CEC2017-F3

CEC2017-F4



CEC2017-F5

	GOA(ON 50 RUNS)			IGOA(ON 50 RUNS)		
	MEAN	VAR	BEST	MEAN	VAR	BEST
F3	6065.2	18.371	6059.7	6064.8	14.025	6059.7
F4	263.9	2.4997E-07	263.9	263.9	1.2158E-05	263.9
F6	1.3409	3.9831E-07	1.3401	1.341	8.7799E-07	1.34
F8	26.487	2.8514E-07	26.487	26.488	1.0512E-06	26.486
F13	359.21	9.1389E-10	359.21	359.21	4.4943E-10	359.21

6.3. Performance Comparison with 6 state-of-the-art algorithms

6.3.1. Comparison on CEC2017 AND CEC2014

CEC 2017 (MEAN ON 50 RUNS)

	GOA	IGOA	SCSO	AOA	FHO	WOA	GWO	COA
F1	6.1052E+06	3961.3	1.1554E+07	3667.7	4.3473E+08	3.2499E+06	1.4546E+07	6.5927E+05
F2	0	0	0	0	0	0	0	0
F3	518.46	304.27	897.22	300.8	4794.2	646.69	785.05	3870.9
F4	406.55	405.52	432.65	407.64	478.79	444.33	433.9	427.16
F5	513.67	516.86	532.42	520.34	541.54	540.25	510.87	520.75
F6	601.22	603.79	610.11	600.59	619.85	627.56	600.87	603.55
F7	733.68	735.5	753.35	731.46	781.06	777.8	730.37	749.15
F8	813.98	817.43	834.5	822.35	836.4	842.3	812.11	824.22
F9	903.34	945.8	1140.2	923.6	1087	1640.5	906.08	1141.5
F10	1479.4	1434.7	1822.9	1515.6	2180	1944.6	1512.6	2313.5
F11	1109.1	1107.9	1130	1115.7	1216.7	1147.2	1112.7	1127.3
F12	1.0431E+05	29835	2.9159E+06	12859	6.5131E+06	3.0195E+06	2.2089E+06	6.0841E+05
13	1696.2	1494.5	14533	2889.9	31082	14157	13736	8060.9
F14	1448.8	1436.8	1805.9	1703.4	1609.2	1606.1	1961.1	2323.9
F15	1555.5	1524.8	1777.3	1602	1970.8	5249.1	2166.8	2506.5
F16	1608.3	1610.5	1713.1	1691.8	1697.6	1753.3	1655	1705.5
F17	1739.7	1736.2	1761.1	1746.6	1778.2	1803.6	1746.2	1761.2
F18	2396.1	2000.3	12067	6735.4	29862	15132	11711	12876
F19	1915.3	1909.8	2351.8	1969.2	2843.1	3144	2274.7	2887.3
F20	2038.9	2042.5	2089.7	2054.5	2092.7	2153	2044	2074.7
F21	0	0	0	0	0	0	0	0
F22	0	0	0	0	0	0	0	0
F23	2666.8	2672.1	2500	2653.1	2538.3	2708.6	2662.2	2500
F24	2560.1	2513.6	2600	2602.8	2612.1	2758.3	2781.5	2600
F25	2900.1	2898.8	2700	2929.3	2974.8	2955.1	2938.4	2700
F26	2782.8	2780.7	2800	3052.9	3039.4	3126.6	3063	2800
F27	3134.8	3133.1	2900	3185.5	3134.1	3231.1	3153.5	2900
F28	3121.6	3110.4	3000	3155.5	3207.8	3222.8	3180.9	3000
F29	3165	3160	3100	3189.3	3202.2	3246.3	3164.5	3100
F30	9870.1	4699.2	4759.1	6290.2	15657	6.8465E+05	12036	3200

CEC 2014 (MEAN ON 50 RUNS)

	GOA	IGOA	P VALUE	SCSO	AOA	FHO	WOA	GWO	COA
F1	300880.2763	66023.80933	1.60E-16	4010100	104180	4650000	8339600	2771200	1758100
F2	3994809.729	6061.320114	7.07E-18	141960	3829.2	850010000	4580400	175790	150490
F3	749.4	448.96	5.84E-12	2974	595.62	6568.2	53319	6498	10242
F4	408.6	403.56	1.80E-09	429.05	407.79	480.14	445.32	433.3	427.71
F5	519.66	519.34	2.71E-15	520.12	520.04	520.39	520.16	520.45	520.31
F6	601.6389591	603.0187046	2.13E-09	604.85	601.87	606.84	607.52	601.65	604.18
F7	701.01	700.51	1.58E-13	700.65	700.2	706.65	701.12	701.13	700.36
F8	805.95	806.73	2.81E-01	829.05	812.8	835.47	836.86	809.8	811.85
F9	914.0825507	916.5901018	3.02E-02	931.06	922.78	939.5	940.4	911.66	934.94
F10	1098.042925	1170.50471	4.76E-03	1662.1	1217.9	2008.7	1635.6	1367.3	1517.4
F11	1602.8	1589.8	7.49E-01	2057	1891.1	2329.2	2117.7	1553.6	1929.8
F12	1200.6	1200.3	8.94E-14	1200.4	1200.5	1201.2	1200.8	1201.2	1200.6
F13	1300.228171	1300.20402	1.03E-01	1300.3	1300.2	1300.8	1300.4	1300.2	1300.3
F14	1400.17366	1400.154276	1.89E-02	1400.4	1400.3	1400.6	1400.3	1400.2	1400.4
F15	1502.375395	1501.937685	5.05E-04	1503.1	1501.4	1511.4	1506.8	1501.9	1502.5
F16	1602.7	1602.8	4.38E-02	1603	1603	1603.5	1603.4	1602.6	1603
F17	2571.771204	2262.940877	1.84E-11	6738.8	4100.9	11150	217630	24648	32564
F18	1885.5	1873.4	1.06E-02	12785	10181	10698	15087	10322	9900.6
F19	1902.1	1902	3.95E-01	1903.5	1901.9	1907.3	1905.8	1902.5	1902.9
F20	2054.044006	2031.389122	3.02E-10	5252.7	2424.6	3034.2	6718.4	5527.1	5995
F21	2461	2294.8	7.12E-11	8503.1	2488.5	7249.2	106580	8552.8	6985.5
F22	2223.9	2220.5	1.58E-07	2275.2	2271.1	2251	2295.7	2252.5	2251
F23	2577.0601	2549.164879	2.99E-05	2500	2629.5	2648.9	2635.8	2631.1	2500
F24	2519.6	2522.8	8.71E-03	2564.5	2526	2551.6	2557.3	2518.9	2578.3
F25	2641.767658	2636.949789	6.96E-03	2697.4	2651.5	2667.4	2696.3	2697.2	2698.5
F26	2700.224484	2700.18401	1.44E-04	2700.3	2700.2	2700.5	2700.4	2700.2	2700.3
F27	2704.418578	2706.411162	1.34E-09	2845.8	2815.8	2756.7	3024	2969.4	2869.1
F28	3179.4	3176.8	9.83E-03	3000	3175.8	3173.8	3313.7	3226.4	3000
F29	3310.464308	3215.387373	1.48E-10	4349.8	3349.2	45250	4573.3	38388	44642
F30	3630.203666	3595.495106	4.83E-02	4638.3	3790.3	4205	5203.1	3961.5	4263.4

7. Conclusion and future work

References

[1] E.-G. Talbi, Metaheuristics: from design to implementation, John Wiley & Sons, 2009.

[2] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, A. H. Gandomi, The arithmetic optimization algorithm, Computer methods in applied mechanics and engineering 376 (2021) 113609.

[3] M. Jamil, X.-S. Yang, A literature survey of benchmark functions for global optimisation problems, International Journal of Mathematical Modelling and Numerical Optimisation 4 (2) (2013) 150–194.

[4] M. Abdel-Basset, L. Abdel-Fatah, A. K. Sangaiah, Chapter 10 - metaheuristic algorithms: A comprehensive review, in: A. K. Sangaiah, M. Sheng, Z. Zhang (Eds.), Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications, Intelligent Data-Centric Systems, Academic Press, 2018, pp. 185–231.

[5] K. Rajwar, K. Deep, S. Das, An exhaustive review of the metaheuristic algorithms for search and optimization: Taxonomy, applications, and open challenges, Artificial Intelligence Review 56 (11) (2023) 13187–13257.

[6] F. Kiani, F. A. Anka, F. Erenel, Pscso: Enhanced sand cat swarm optimization inspired by the political system to solve complex problems, Advances in Engineering Software 178 (2023) 103423.
[7] D. Whitley, S. Rana, J. Dzubera, K. E. Mathias, Evaluating evolutionary algorithms, Artificial intelligence 85 (1-2) (1996) 245–276.
[8] J. H. Holland, Genetic algorithms, Scientific american 267 (1) (1992) 66–73.
[9] D. Simon, Biogeography-based optimization, IEEE transactions on evolutionary computation 12 (6) (2008) 702–713.
[10] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, Journal of global optimization 11 (1997) 341–359.
[11] F. Glover, Tabu search: A tutorial, Interfaces 20 (4) (1990) 74–94.
[12] V. Hayyolalam, A. A. P. Kazem, Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems, Engineering Applications of Artificial Intelligence 87 (2020) 103249.
[13] R.-J. Kuo, F. E. Zulvia, The gradient evolution algorithm: A new metaheuristic, Information Sciences 316 (2015) 246–265.
[14] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, Gsa: a gravitational search algorithm, Information sciences 179 (13) (2009) 2232–2248.
[15] F. A. Hashim, K. Hussain, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems, Applied intelligence 51 (2021) 1531–1551.
[16] J. L. J. Pereira, M. B. Francisco, C. A. Diniz, G. A. Oliver, S. S. Cunha Jr, G. F. Gomes, Lichtenberg algorithm: A novel hybrid physics-based meta-heuristic for global optimization, Expert Systems with Applications 170 (2021) 114522.
[17] F. A. Hashim, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, S. Mirjalili, Henry gas solubility optimization: A novel physics-based algorithm, Future Generation Computer Systems 101 (2019) 646–667.
[18] F. A. Hashim, R. R. Mostafa, A. G. Hussien, S. Mirjalili, K. M. Sallam, Fick's law algorithm: A physical law-based algorithm for numerical optimization, Knowledge-Based Systems 260 (2023) 110146.
[19] M. Abdel-Basset, R. Mohamed, S. A. A. Azeem, M. Jameel, M. Abouhawwash, Kepler optimization algorithm: A new metaheuristic algorithm inspired by kepler's laws of planetary motion, Knowledge-Based Systems 268 (2023) 110454.
[20] H. Shah-Hosseini, Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation, International journal of computational science and engineering 6 (1-2) (2011) 132–140.
[21] S.-A. Ahmadi, Human behavior-based optimization: a novel metaheuristic approach to solve complex optimization problems, Neural Computing and Applications 28 (Suppl 1) (2017) 233–244.
[22] S. J. Mousavirad, H. Ebrahimpour-Komleh, Human mental search: a new population-based metaheuristic optimization algorithm, Applied Intelligence 47 (2017) 850–887.
[23] I. Matoušová, P. Trojovskỳ, M. Dehghani, E. Trojovská, J. Kostra, Mother optimization algorithm: A new human-based metaheuristic approach for solving engineering optimization, Scientific Reports 13 (1) (2023) 10312.
[24] M. Dehghani, E. Trojovská, P. Trojovskỳ, A new human-based metaheuristic algorithm for solving optimization problems on the base of simulation of driving training process, Scientific reports 12 (1) (2022) 9924.
[25] Q. Askari, I. Younas, M. Saeed, Political optimizer: A novel socio-inspired meta-heuristic for global optimization, Knowledge-based systems 195 (2020) 105709.
[26] R. V. Rao, R. V. Rao, Teaching-learning-based optimization algorithm, Springer, 2016.
[27] N. Moosavian, B. K. Roodsari, Soccer league competition algorithm: A novel meta-heuristic algorithm for optimal design of water distribution networks, Swarm and Evolutionary Computation 17 (2014) 14–24.
[28] S. Balochian, H. Baloochian, Social mimic optimization algorithm and engineering applications, Expert Systems with Applications 134 (2019) 178–191.
[29] T. Rahkar Farshi, Battle royale optimization algorithm, Neural Computing and Applications 33 (4) (2021) 1139–1157.
[30] X.-S. Yang, M. Karamanoglu, Nature-inspired computation and swarm intelligence: a state-of-the-art overview, Nature-Inspired Computation and Swarm Intelligence (2020) 3–18.
[31] A. Kumar, M. Nadeem, H. Banka, Nature inspired optimization algorithms: a comprehensive overview, Evolving Systems 14 (1) (2023) 141–156.
[32] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95-international conference on neural networks, Vol. 4, IEEE, 1995, pp. 1942–1948.
[33] M. Azizi, S. Talatahari, A. H. Gandomi, Fire hawk optimizer: A novel metaheuristic algorithm, Artificial Intelligence Review 56 (1) (2023) 287–363.
[34] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, Advances in engineering software 69 (2014) 46–61.
[35] S. Mirjalili, A. Lewis, The whale optimization algorithm, Advances in engineering software 95 (2016) 51–67.
[36] R. Salgotra, U. Singh, A novel bat flower pollination algorithm for synthesis of linear antenna arrays, Neural Computing and Applications 30 (2018) 2269–2282.

[37] M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, IEEE computational intelligence magazine 1 (4) (2006) 28–39.
[38] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, Advances in engineering software 114 (2017) 163–191.
[39] J. O. Agushaka, A. E. Ezugwu, L. Abualigah, Gazelle optimization algorithm: a novel nature-inspired metaheuristic optimizer, Neural Computing and Applications 35 (5) (2023) 4099–4131.
[40] P. Trojovskỳ, M. Dehghani, Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications, Sensors 22 (3) (2022) 855.
[41] F. Kiani, A. Rad, M. Sis, A. Kut, A. Alpkocak, Eear: an energy effective-accuracy routing algorithm for wireless sensor networks, Life Science Journal 10 (2) (2013) 39–45.
[42] A. Guzmán-Ponce, J. R. Marcial-Romero, R. M. Valdovinos, R. Alejo, E. Granda-Gutiérrez, A metaheuristic algorithm to face the graph coloring problem, in: International Conference on Hybrid Artificial Intelligence Systems, Springer, 2020, pp. 195–208.
[43] F. Kiani, A novel channel allocation method for time synchronization in wireless sensor networks, International Journal of Numerical Modelling: Electronic Networks, Devices and Fields 29 (5) (2016) 805–816.
[44] M. A. Silgu, H. B. Celikoglu, Clustering traffic flow patterns by fuzzy c-means method: some preliminary findings, in: Computer Aided Systems Theory–EUROCAST 2015: 15th International Conference, Las Palmas de Gran Canaria, Spain, February 8-13, 2015, Revised Selected Papers 15, Springer, 2015, pp. 756–764.
[45] F. Altiparmak, B. Dengiz, A. E. Smith, Optimal design of reliable computer networks: A comparison of metaheuristics, Journal of heuristics 9 (2003) 471–487.
[46] J. M. Lanza-Gutierrez, J. A. Gomez-Pulido, Assuming multiobjective metaheuristics to solve a three-objective optimisation problem for relay node deployment in wireless sensor networks, Applied Soft Computing 30 (2015) 675–687.
[47] A. Seyyedabbasi, F. Kiani, Sand cat swarm optimization: A nature-inspired algorithm to solve global optimization problems, Engineering with Computers 39 (4) (2023) 2627–2651.