BenchCouncil Transactions on Benchmarks, Standards and Evaluations 5 (2025) 100250

Contents lists available at ScienceDirect

BenchCouncil Transactions on Benchmarks, Standards and Evaluations

journal homepage: www.keaipublishing.com/en/journals/benchcouncil-transactions-onbenchmarks-standards-and-evaluations/

Full Length Article

An adaptive opposite slime mold feature selection algorithm for complex optimization problems☆

Elsayed Badr [a,b,c], Mostafa Abdullah Ibrahim [b], Diaa Salama [b,d], Mohammed ElAffendi [e], Abdelhamied A Ateya [e,f,*], Mohamed Hammad [e,g], Alaa Yassin [a,b]

[a] Faculty of Computers and Artificial Intelligence, Misr University for Science & Technology, Egypt
[b] Faculty of Computers and Artificial Intelligence, Benha University, Egypt
[c] The Egyptian School of Data Science (ESDS), Benha, Egypt
[d] Faculty of Computers and Information, Misr International University, Egypt
[e] EIAS Data Science Lab, College of Computer and Information Sciences, Prince Sultan University, Riyadh, 11586, Saudi Arabia
[f] Department of Electronics and Communications Engineering, Zagazig University, Zagazig 44519, Egypt
[g] Department of Information Technology, Faculty of Computers and Information, Menoufia University, Shibin El Kom, 32511, Egypt

ARTICLE INFO

Keywords:
Slime mold algorithm (SMA)
Optimization
Adaptive opposite SMA (AOSMA)
Enhanced SMA (ESMA)
Modified SMA (MSMA)

ABSTRACT

The slime mould algorithm (SMA) has recently emerged as a soaring metaheuristic strategy to function optimization problems due to its solid exploration-exploitation balance that enables it to converge efficiently towards high-quality solutions. In spite of its broader applications, however, there remain areas where the algorithm is constrained in diversified exploration and the scope of its exploitation mechanisms. In bridging these loopholes, this work introduces a new variant known as the adaptive opposition SMA (AOSMA). AOSMA involves an adaptive opposition-based learning (OBL) method, which learns online how to add opposition-based solutions at the iteration process to enhance exploration abilities and avoid premature convergence. The adaptive policy enables the algorithm to escape local optima more effectively by occasionally generating alternative candidate solutions. Additionally, for the sake of increased exploitation, AOSMA also incorporates a plan in which the randomly selected search agent is progressively replaced with the current best-performing agent during position updating. The replacement process increases the focus of the algorithm towards prospective regions of the search space, and thus it converges more quickly towards the global optimum. The implemented AOSMA was exhaustively validated with both qualitative and quantitative measures in terms of thirteen rigorously proven benchmark test functions involving a variety of unimodal, multimodal, and composite landscapes to test its optimization ability extensively. Comparative tests on a collection of state-of-the-art metaheuristic algorithms confirmed that AOSMA consistently produces higher or highly comparable performances across a variety of problem instances. The experimental results confirm the robustness, adaptability, and improved search ability of the algorithm, highlighting its potential as an efficient optimization method for complex real-world problems. With the efficient fusion of adaptive exploration and improved exploitation, AOSMA provides a vital contribution to the field of research into swarm intelligence and metaheuristic optimization.

1. Introduction

Maximum utilization of resources is a basic requirement in numerous scientific, engineering, and industrial applications. Because of the increasing complexity of real-world problems and the scarcity of resources, optimization has become a necessity for ensuring the highest performance, cost-effectiveness, and operating efficiency. Optimization problems can be broadly categorized into deterministic and stochastic approaches, depending upon the type of problem, each with its advantages and methodological strategies depending upon the problem landscape [1]. Deterministic optimization methods, such as linear and nonlinear programming, rely extensively on derivative information. These methods work best when there is structured problem formulation and the search space is smooth or linear. Deterministic approaches excel

☆ Peer review under the responsibility of The International Open Benchmark Council.

* Corresponding author.
E-mail address: aateya@psu.edu.sa (A.A. Ateya).

https://doi.org/10.1016/j.tbench.2025.100250
Received 10 July 2025; Received in revised form 8 December 2025; Accepted 9 December 2025
Available online 13 December 2025

Table 1
Comparison between the proposed approach and closely related prior works that already combine OBL with SMA (e.g., [10,13,16]).

Algorithm	Core Idea	Strengths	Limitations	Relation to Present Work
SCA (Sine Cosine Algorithm) [10]	Uses sine and cosine mathematical functions to update positions and control exploration/ exploitation.	Simple, easy to implement, competitive on many benchmarks.	Sensitive to parameter tuning, risk of premature convergence on multimodal problems.	Illustrates novel update dynamics; lacks adaptive decision strategies.
New Meta-heuristic [13]	Introduces a new population-based global optimizer emphasizing exploration ability.	Demonstrates strong global search potential, competitive performance.	Does not include adaptive control; balance between exploration and exploitation can weaken.	Shows effectiveness of new frameworks but highlights need for adaptive balancing.
SMA Survey [16]	Comprehensive review of SMA and its variants (chaotic maps, OBL, hybrids).	Demonstrates SMA's flexibility, wide applications, and many successful variants.	Variants often apply OBL blindly or intensification operators without systematic rules; no consensus on when to switch.	Identifies gap: absence of adaptive OBL decision + replacement integration.
Proposed AOSMA	Combines adaptive OBL decision strategy with a replacement mechanism.	Selective exploration, improved exploitation, faster convergence, robust performance.	Slightly higher computational cost.	Directly addresses the gap identified in SMA literature by integrating adaptive exploration control with exploitation enhancement.

Table 2
Difference between this paper's combination of OBL and SMA and the related work.

Algorithm	Core Idea	Enhancement Strategy	Difference from AOSMA
OJESMA (W. C. Wang et al. [13])	Improve SMA using hybrid strategies	Combines equilibrium optimizer, joint opposite selection, and OBL	Depends on external optimizers and hybridization, whereas AOSMA focuses on integrated OBL with adaptive exploitation
Adaptive SMA (M. K. Naik et al. [10])	Enhance OBL application adaptively	Uses an adaptive mechanism to decide OBL usage; improves exploitation by replacing random agents with the best agent	Focuses mainly on OBL decision-making and selective replacement; AOSMA instead integrates OBL with adaptive exploitation for broader balance
AOSMA (Proposed)	Advance slime mould optimization	Combines OBL and adaptive exploitation techniques to balance exploration and exploitation dynamically	Unlike OJESMA (hybrid) or Adaptive SMA (OBL-focused), AOSMA unifies OBL and adaptive exploitation into a single integrated framework

in linear or weakly nonlinear spaces but experience difficulties when confronted with highly nonlinear, multi-modal, or discontinuous spaces. In such challenging cases, their dependence on gradient information can be a limiting aspect, conditioning them to get stuck in local optima or not to converge at all [2].

Stochastic optimization techniques, by contrast, never employ direct gradient information. This kind of algorithm forms the foundation for metaheuristic optimization methods. Metaheuristics are designed to provide rigorous, general-purpose solutions that can be applied to a large class of complex problems without specific knowledge of the mathematical form [3]. Their inherent advantages include simplicity, flexibility, independence of gradients, and less reliance on initial candidate solutions, traits that have driven their popularity and dramatic rise over the past few years. Metaheuristic algorithms have proven outstandingly successful in optimization problem-solving in numerous applications, from engineering design and scheduling to machine learning, hyperparameter tuning, and biomedical image analysis. They are largely inspired by natural, physical, or social processes and thus form a vast and growing taxonomy. Metaheuristic algorithms fall under four broad classes: swarm intelligence-based, evolutionary-based, physics-based, and human-inspired algorithms [4].

These nature-inspired metaheuristic algorithms possess specific strengths as they emulate some adaptive and intelligent behavior from nature, resulting in very efficient tools. An innovative nature-inspired optimization algorithm is the slime mould algorithm (SMA), which is based on the nature-inspired foraging behavior and dynamic oscillatory movement exhibited by slime moulds (Physarum polycephalum) [5]. SMA has been demonstrated to possess a strong capability to handle extremely complex, high-dimensional, and multi-modal optimization problems. Due to its strong global search capability and excellent resistance to local optima, SMA has been effectively applied to many applications in machine learning, engineering design, feature selection, and image processing [6].

Although it has an outstanding performance, SMA also faces some disadvantages, particularly concerning its exploration and exploitation. There were findings indicating that SMA often has acceptable exploration of the search space. However, its capacity for exploitation is still in need of enhancement to achieve faster convergence and higher-quality solutions. To address such limitations, numerous methods have been proposed in recent years. As documented in the literature [5], the two most significant improvement strategies that have emerged are hybridization, where SMA is combined with other metaheuristics to leverage complementary strengths, and algorithmic refinement, which involves modifying or improving SMA's basic equations and mechanisms to improve performance.

A case in point is Chen et al. [7], which proposed the RCLSMAOA algorithm, a hybrid metaheuristic that combined the SMA and arithmetic optimization algorithm (AOA). The hybridization approach enhanced the convergence rate and solution accuracy, showing the potential of blended algorithmic solutions to counter weaknesses in individual approaches. Building on such advancements, this work proposes an adaptive variant of SMA in the guise of the adaptive opposition SMA (AOSMA). AOSMA employs an adaptive opposition-based learning (OBL) process to adaptively determine when to inject opposition solutions, enhancing the ability of the algorithm to explore without precipitating premature convergence. In addition, a novel exploitation enhancement method is incorporated through automatically substituting a randomly selected search agent with the best-performing agent at the position update phase, focusing the search on promising regions.

The remainder of this paper has the following structure: Section 2 provides a comprehensive review of existing literature and recent additions to the SMA; Section 3 provides a detailed methodology and the AOSMA model suggested; Section 4 provides experimental findings and comparative analyses; and finally, Section 5 summarizes the research with conclusions and future study directions.

2. Related works

Metaheuristic algorithms have been of much importance as efficient tools to tackle nonlinear and complex optimization issues where

Table 3
Properties of the considered standard benchmark functions.

Function	Function	D	F min	Range
F1	$\sum_{i=1}^{n} X_i^2$	30	0	[100,-100]
F2	$\sum_{i=1}^{n} X_i^2 + \prod_{i=1}^{n} \lvert X_i \rvert$	30	0	[10,-10]
F3	$\sum_{i=1}^{n} \left(\sum_{j=1}^{i} X_j \right)^2$	30	0	[100,-100]
F4	$\max i \{ \lvert x \rvert,\ 1 \le i \le n \}$	30	0	[100,-100]
F5	$\sum_{i-1}^{n-1} \left(100(X_{i+1} - X_i)^2 + (X_i - 1)^2 \right)$	30	0	[30,-30]
F6	$\sum_{i=1}^{n} (X_i + 0.5)^2$	30	0	[100,-100]
F7	$\sum_{i=1}^{n} \left(X_i^4 + rand\ (0,1) \right)$	30	0	[1.28,-1.28]
F8	$\sum_{i=1}^{n} - X_i^2 \sin \sqrt{\lvert X_i \rvert}$	30	-418.98 N	[500,-500]
F9	$\sum_{i=1}^{n} \left[X_i^2 - 10\cos(2\pi X_i) + 10 \right]$	30	0	[5.12,-5.12]
F10	$-20exp\left(-0.2\sqrt{\sum_{i=1}^{n} x_i^2} \right) - exp\left(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e$	30	0	[32,-32]
F11	$\frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	30	0	[600,-600]
F12	$\frac{\pi}{n} \left\{ 10\ sin(\pi y_1) + \sum_{i=1}^{n-1} (y_{i+1})^2 \left[1 + 10\sin^2(\pi y_{i+1}) \right] + (y_n - 1)^2 \right\} + \sum_{i=1}^{n} u(x_i,\ a,\ k,\ m)$, $y_i = 1 + \frac{x_i + 1}{4}$, $u(x_i,\ a,\ k,\ m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \le x_i < a \\ k(x_i - a)^m & x_i < -a \end{cases}$	30	0	[50,-50]
F13		30	0	[50,-50]

traditional deterministic approaches are inefficient. Among them, swarm intelligence-based approaches such as PSO, ACO, and ABC have been found very successful in traversing extensive and complicated search spaces. Requested by the natural behavior, these algorithms can efficiently explore and exploit in a bid to provide efficient solutions to computationally complex real-world problems. However, the continually rising global optimization problems have motivated scholars to seek even more flexible and hybridized methods to handle issues like premature convergence and low convergence rates [8,9].

SMA, though showing its prowess in many applications, is underutilized with respect to its potential, primarily dynamic adaptability and enhanced exploration mechanisms. Recent developments in SMA are mainly directed towards hybridization with other approaches or modifications of update equations in the central part for enhancing global search capability and rate of convergence. With these efforts, issues such as maintaining diversity in the search space and local optima avoidance still exist. This has motivated the development of the AOSMA with the objective of integrating OBL and adaptive exploitation techniques for further advancing the state-of-the-art in slime mould-based optimization [10]. This section provides the related studies to introduce the novelty of the proposed approach better.

SMA, inspired by biological behavior, is shown to be highly effective for solving challenging stochastic optimization problems. Its ability to imitate slime mould search patterns offers a competitive edge in global optimization. However, it struggles with premature convergence and suboptimal solutions in complex scenarios. To resolve this, Yang et al. [11] proposed a multi-chaotic local operator that is introduced into the feedback system to enhance local exploration through chaotic perturbations. In [12], the authors first reviewed and analyzed numerous advanced variants of the SMA, providing a comprehensive summary, classification, and discussion of their mechanisms and prospects. Secondly, they categorized the application areas of SMA, examining its functions, current development status, and existing limitations. The literature review shows that SMA outperforms several well-known metaheuristic algorithms in terms of convergence speed and accuracy across various benchmark functions and practical optimization tasks.

Wang et al. [13] suggested the OJESMA approach, an enhanced SMA. Through the combination of equilibrium optimizer, joint opposite selection, and OBL, OJESMA enhances the algorithm's performance. The study involved applying nonparametric tests (Friedman and Wilcoxon) to evaluate optimization performance on 10 CEC2020 benchmark functions and 29 CEC2017 functions. The outcomes of the non-parametric tests and the experiment demonstrate that OJESMA performs better in terms of stability, convergence, and optimization accuracy. The authors also ran optimization tests on the variable index Muskingum and six engineering challenges to confirm the algorithm's efficacy. In [14], the authors presented an improved algorithm that combines an adaptive search operator strategy based on the Cauchy inverse cumulative distribution (QCMSMA) with Bloch sphere-based elite population initialization. The Wilcoxon rank-sum test and Friedman ranking analysis were used for statistical evaluations, along with comparisons against a number of popular optimization techniques.

Houssein et al. [15] proposed another work that uses the SMA adaptive guided differential evolution algorithm (SMA-AGDE) to address some inherent weaknesses of the original SMA. By employing AGDE's mutation strategy within it, the hybrid approach enhances local search power, population diversity, and prevention of getting trapped in local minima. Comparative research involving a range of well-established, newly developed, and high-performance metaheuristics (such as BBO, GSA, TLBO, HHO, MRFO, CMA-ES, and the simple SMA) revealed that SMA-AGDE was significantly better than its alternatives in all instances of problems, being first in different problem contexts. The research verifies that SMA-AGDE is an effective and generic optimization platform.

In [5], the authors examined the SMA from various optimization perspectives. The algorithm utilizes a specialized mathematical framework that imitates biological wave propagation using adaptive weighting, introducing several innovative features to enhance performance. This design enables an effective balance between exploration and exploitation, guiding the search efficiently toward optimal solutions. In [16], the authors addressed a four-objective optimization problem in the construction industry by proposing a hybrid model called AOSMA, which integrates OBL with the SMA. To showcase the effectiveness of the proposed approach, two real-world construction case studies were used,

Table 4
Fitness values of the proposed AOSMA and other algorithms.

Function	Run	f_{min}	Sinusoidal SMA	Circle SMA	Singer SMA	Tent SMA	Logistic SMA	Sine SMA	Iterative SMA	Mouse SMA	AOSMA
F1	1	0	0	0	2.5852	0	0	0	0	0	0
	2		0	0	1.21E-42	0	0	0	0	0	0
	3		0	0	9.68E-50	0	0	0	0	0	0
	4		0	0	1.73E-09	0	1.6446e-317	0	0.00E+00	0	0
	5		0	0	2.44E-18	0	0	0	0	0	0
F2	1	0	5.03E-199	3.18E-215	6.66E-05	3.72E-158	2.71E-175	1.30E-199	3.72E-158	1.10E-240	0
	2		5.09E-182	1.14E-270	1.49E-10	1.25E-215	1.73E-288	1.01E-166	1.25E-215	5.44E-163	0
	3		1.69E-164	9.93E-194	0.21205	2.40E-303	4.66E-182	2.03E-177	2.40E-303	7.60E-245	0
	4		3.50E-220	4.27E-218	2.15E-24	2.34E-185	2.19E-213	8.41E-173	2.34E-185	2.29E-187	0
	5		1.02E-177	3.55E-209	9.99E-27	1.20E-253	4.21E-264	7.49E-264	1.20E-253	6.81E-276	0
F3	1		0.00E+00	0.00E+00	0.0078075	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0
	2		0.00E+00	2.9644e-32	1.42E-77	0.00E+00	5.2707e-319	0.00E+00	0.00E+00	0.00E+00	0
	3		0.00E+00	0.00E+00	191.5055	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0
	4		0.00E+00	0.00E+00	1.0838	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0
	5		0.00E+0000	0.00E+00	3.3663	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0
F4	1	0	2.05E-189	6.41E-192	1.16E-206	1.38E-231	7.67E-222	3.08E-166	1.38E-231	2.92E-197	0
	2		5.43E-182	4.65E-192	1.23E-244	1.27E-211	2.81E-198	9.45E-204	1.27E-211	6.01E-158	0
	3		7.61E-165	1.22E-242	5.90E-237	3.33E-206	3.36E-225	1.72E-197	3.33E-206	1.37E-238	0
	4		3.77E-236	8.40E-174	8.32E-255	2.63E-177	1.18E-180	1.85E-150	2.63E-177	3.81E-192	0
	5		8.32E-182	1.88E-189	1.19E-220	4.45E-239	5.55E-214	1.33E-174	4.45E-239	1.50E-174	0
F5	1	0	5.0595	1.0302	0.15428	28.2287	0.63358	3.1302	28.2287	0.06720	0.02504
	2		2.7025	0.09351	16.1099	0.33496	1.4679	5.6199	0.33496	0.55468	26.8342
	3		2.9999	0.2898	16.1381	28.1521	1.2347	0.51309	28.1521	8.9787	0.03030
	4		0.22394	1.0852	0.01094	3.2781	2.1888	1.2713	3.2781	2.2301	0.02494
	5		28.2597	0.15391	15.3578	4.42	1.2908	28.1758	4.42	2.9323	0.50422
yyy	1	0	0.0028185	0.0032071	0.21129	0.0079421	0.0046937	0.0035018	0.0079421	0.0083993	5.70E-05
	2		0.0078915	0.005237	4.684	0.0039248	0.0092994	0.0040872	0.0039248	0.0043632	6.49E-05
	3		0.0082295	0.0070498	0.65814	0.0071398	0.0061981	0.0031065	0.0071398	0.0051079	3.75E-05
	4		0.0026498	0.00038786	1.2805	0.0031112	0.0051964	0.0017886	0.0031112	0.0053033	4.10E-05
	5		0.0069716	0.0023814	1.10E-05	0.0041485	0.0068288	0.0039413	0.0076717	0.013709	6.01E-05
F7	1	0	0.00017136	0.00017765	0.04975	0.0003222	4.44E-05	3.47E-05	0.00032227	0.0001375	0.0001596
	2		3.16E-05	0.00019332	0.0096608	0.00026656	0.00014147	0.00032235	0.00029389	0.00028793	0.00015071
	3		0.0001055	0.00015583	0.015933	0.0005641	0.00021668	8.67E-05	0.0005641	0.0003751	4.72E-05
	4		0.0003279	8.12E-06	0.0474	0.0004275	0.00015481	0.0001298	0.00042759	0.0001863	6.01E-06
	5		0.0001248	2.68E-05	0.0030962	0.0001135	5.47E-05	0.0003885	0.00011354	0.0005044	2.88E-06
F8	1	-12,569	-12,568.589	-12,569.337	-12,563.69	-12,569.426	-12,569.4295	-12,568.447	-12,569.426	-12,569.405	-12,569.483
	2		-12,569.088	-12,569.364	-12,554.71	-12,569.422	-12,569.1176	-12,569.301	-12,569.422	-12,569.255	-12,569.483
	3		-12,569.141	-12,569.423	-12,569.48	-12,569.465	-12,569.087	-12,568.318	-12,569.465	-12,568.722	-12,569.483
	4		-12,568.910	-12,568.803	-12,567.72	-12,569.135	-12,568.947	-12,568.002	-12,569.135	-12,569.258	-12,569.483
	5		-12,568.991	-12,569.040	-12,549.79	-12,568.748	-12,568.687	-12,569.473	-12,568.748	-12,569.313	-12,569.483
F9	1	0	0	0	0.05097	0	0	0	0	0	0
	2		0	0	0	0	0	0	0	0	0
	3		0	0	0	0	0	0	0	0	0
	4		0	0	0	0	0	0	0	0	0
	5		0	0	0.02342	0	0	0	0	0	0
F10	1	0	8.88E-16	8.88E-16	0.59841	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16
	2		8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16
	3		8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16
	4		8.88E-16	8.88E-16	9.73E-06	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16
	5		8.88E-16	8.88E-16	3.65E-10	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16
F11	1	0	0.00E+00	0.00E+00	1.02E-07	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	2		0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	3		0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	4		0.00E+00	0.00E+00	0.8752	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	5		0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F12	1	0	0.00019093	7.38E-05	0.0027443	0.0021737	0.011757	0.0010932	0.0021737	0.0036851	8.61E-06
	2		0.00060494	0.0034431	0.0001773	0.0014124	0.0010992	0.00011355	0.0014124	0.00056797	8.36E-06
	3		0.0012813	4.07E-05	0.0083147	0.0033076	0.001464	0.0027819	0.0058759	0.00070109	0.00059161
	4		0.0032576	0.0019526	0.0035314	0.0044847	0.012298	0.0015544	0.003753	0.00024023	0.00058589
	5		0.00030893	0.011637	0.067425	0.007021	4.36E-05	0.0093717	0.010762	0.000781	0.0032647
F13	1	0	0.0051604	0.0013586	0.30418	0.0017013	0.0060032	0.013291	0.0062827	0.010571	6.34E-05
	2		0.001026	0.0026104	0.0002713	0.0010591	0.0015065	0.00015428	0.0030661	0.0037858	3.53E-05
	3		0.004871	0.0017761	0.026338	0.0026932	0.0019489	0.0020771	0.0024803	0.0033837	0.087591
	4		0.00084449	0.0028194	0.0071279	0.005576	0.0042496	0.0010457	0.00012182	0.0065913	0.00021984
	5		0.0071283	0.0058296	0.0004936	0.0065373	0.0079408	0.0001381	0.00015165	0.018941	8.65E-05
	No. of functions reaches 0		**4**	3	0	**4**	2	**4**	**4**	**4**	**6**

where the objectives were evaluated, and the optimal solutions were represented through Pareto fronts. OBL is occasionally introduced to improve the algorithm's ability to explore the search space more effectively.

Naik et al. [10] proposed an adaptive technique for deciding whether to apply OBL. Furthermore, the algorithm improves exploitation by replacing a randomly selected search agent with the best-performing one during the position update process. In [17], the authors proposed ESMA, an improved Slime Mould Algorithm that fuses several techniques to enhance performance. The inclusion of Lévy flights and

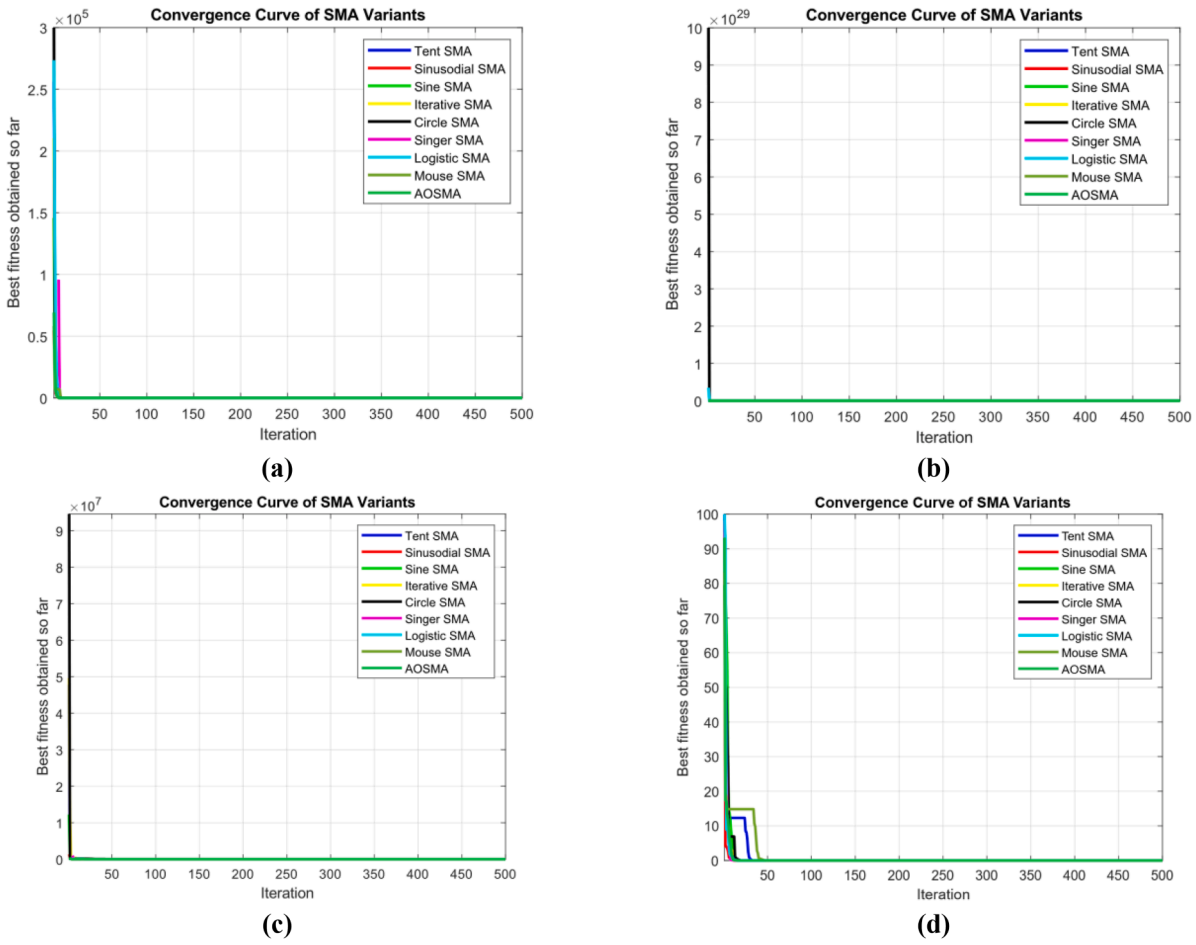

Fig. 1. (a) Convergence curve for f1, (b) Convergence curve for f2, (c) Convergence curve for f3, (d) Convergence curve for f4 obtained by 9 SMA versions for benchmark functions (500 iterations).

selective averaging during the exploration stage enhances adaptability, while a dynamic lens learning method helps reposition the elite solution, steering the swarm toward optimal regions. To this end, the proposed study and the previously introduced literature share the goal of preventing the algorithm from becoming caught in local optima. The difference is in the method utilized. For example, in [14], dynamic random search techniques improve the algorithm's search efficiency. In contrast, AOSMA reduces convergence by employing opposite-based learning and an adaptive decision technique.

2.1. Difference between the proposed combination of OBL and SMA and the related work

Firstly, during the application stage of OBL, the most common point of difference is when OBL is applied within the algorithm's lifecycle. Starting in the initialization phase only: Some studies apply OBL only to the initial population. An "opposite" population is generated, and the best members from both the original and opposite populations are selected to form a more diverse and high-quality starting point. Followed by initialization and generational update, other papers apply OBL not only at the start but also during the iterative process of the algorithm. After the population is updated in a given iteration, OBL can be used to generate opposite solutions for the new population. This helps to prevent stagnation and escape local optima. Finally, the condition-based application: A more sophisticated approach is to apply OBL based on certain conditions, such as population stagnation or slow convergence. This ensures the more computationally intensive OBL operation is only used when needed. Tables 1 and 2 summarize the key comparisons and the novelty of the proposed work compared to the existing literature.

As demonstrated in Ref. [18], AOSMA can be used to address real-world issues described in the abstract. These issues can also be included as benchmark instances. For instance, in (18), the traditional SMA is combined with two more search techniques. While the second uses the Lévy flight distribution (LFD), a real-world problem, to enhance exploration in the early phases and exploitation in the latter stages of the search process, the first uses opposition-based learning (OBL) to speed up convergence.

3. Proposed model

By combining OBL and adaptive control mechanisms, AOSMA significantly improves upon the original SMA. This improves the algorithm's convergence speed, solution diversity, and fitness value in challenging optimization tasks. This section introduces the proposed AOSMA version. The majority of SMA variations with OBL either apply it blindly at every iteration or concentrate just on exploitation without an adaptive rule, which makes it easy to identify the specific research need in the suggested strategy. None combines a straightforward replacement technique with an adaptive OBL choice. This work introduces both to close that gap.

In terms of conceptual analysis, AOSMA enhances the standard SMA by incorporating adaptive opposition-based learning, which generates opposite candidate solutions during the search to maintain diversity and achieve a more effective balance between exploration and exploitation. By contrast, OJESMA combines the Equilibrium Optimizer with joint opposite selection and opposition-based learning, forming a more

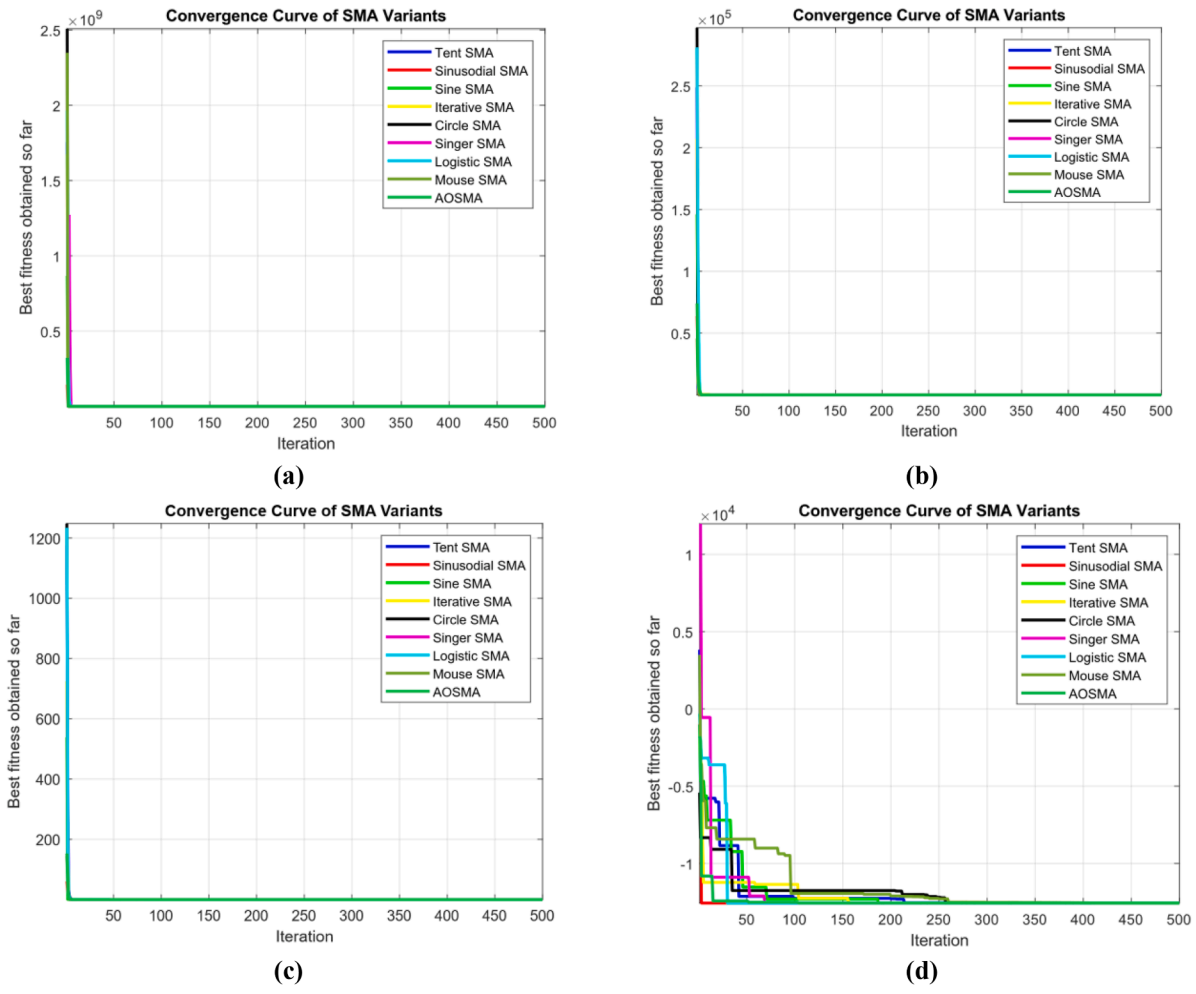

Fig. 2. (a) Convergence curve for f5, (b) Convergence curve for f6, (c) Convergence curve for f7, (d) Convergence curve for f8 obtained by 9 SMA versions for benchmark functions (500 iterations).

advanced hybrid framework that improves convergence stability and accelerates optimization. QCMSMA introduces a Cauchy-based adaptive search operator alongside Bloch sphere-based elite population initialization, thereby boosting exploration capability and enhancing population diversity from the outset. Meanwhile, SMA-AGDE integrates SMA with adaptive guided differential evolution, leveraging DE mutation strategies to strengthen local search performance, foster diversity, and reduce the likelihood of premature convergence.

The fitness function of an optimization or evolutionary problem, written as f(x), is a statistic that assesses how well a particular solution (represented by x) performs or how close it is to achieving the intended goals. Assigning a numerical score (fitness value) to every possible solution is a fundamental part of evolutionary and genetic algorithms; higher scores generally signal better performance and are more likely to be chosen for subsequent generations.

3.1. Mathematical formulation of the AOSMA

Assume that the search space contains N slime mould agents operating within a search space bounded by an upper limit (UB) and a lower limit (LB). The i^{th} slime mould's location in a d-dimensional search space is given by $X_i = (X_i^1, X_i^2, \ldots, X_i^d)$, where i ranges from 1 to N. Its fitness, also referred to as odor, is denoted by $f(X_i)$. Therefore, at iteration t, the positions and fitness values of the **N** slime mould agents at the current iteration **t** can be represented as follows:

$$X(t) = \begin{bmatrix} x_1^1 & x_1^2 & \ldots & x_1^d \\ x_2^1 & x_2^2 & \ldots & x_2^d \\ \vdots & \vdots & \vdots & \vdots \\ x_N^1 & x_N^2 & \ldots & x_N^d \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_N \\ x_N \end{bmatrix} \forall\, N \in \mathbb{R} \tag{1}$$

$$f\,(X) = [f\,(X_1),\, f\,(X_2),\, f\,(X_3), \cdots, f\,(X_N)] \;\forall\, N \in \mathbb{R} \tag{2}$$

The position of slime mould for the subsequent iteration $(t + 1)$ in SMA is updated using Eq. (3).

$$X(t+1) = \begin{cases} X_{LB}(t) + V_b(W.X_A(t) - X_B(t), & if\ r_1 \geq \delta\ and\ r_2 < p_i \\ V_c.X_i(t), & if\ r_1 \geq \delta\ and\ r_2 \geq p_i \\ rand.\ (UB - LB) + LB, & if\ r_1 < \delta \end{cases} \quad \forall\, i \in [1, N] \tag{3}$$

where X_{LB} refers to the best-performing slime mould in the current iteration, X_A and X_B are two randomly chosen individuals from the current population, W acts as the weighting factor, V_b and V_c are random velocity terms drawn from uniform distributions over the ranges [–b, b] and [–c, c], and r_1 and r_2 are random values in the interval [0, 1]. The parameters b and c are updated at each iteration t using the following equations.

$$b = \text{arctanh}\left(-\left(\frac{t}{T} + 1\right)\right) \tag{4}$$

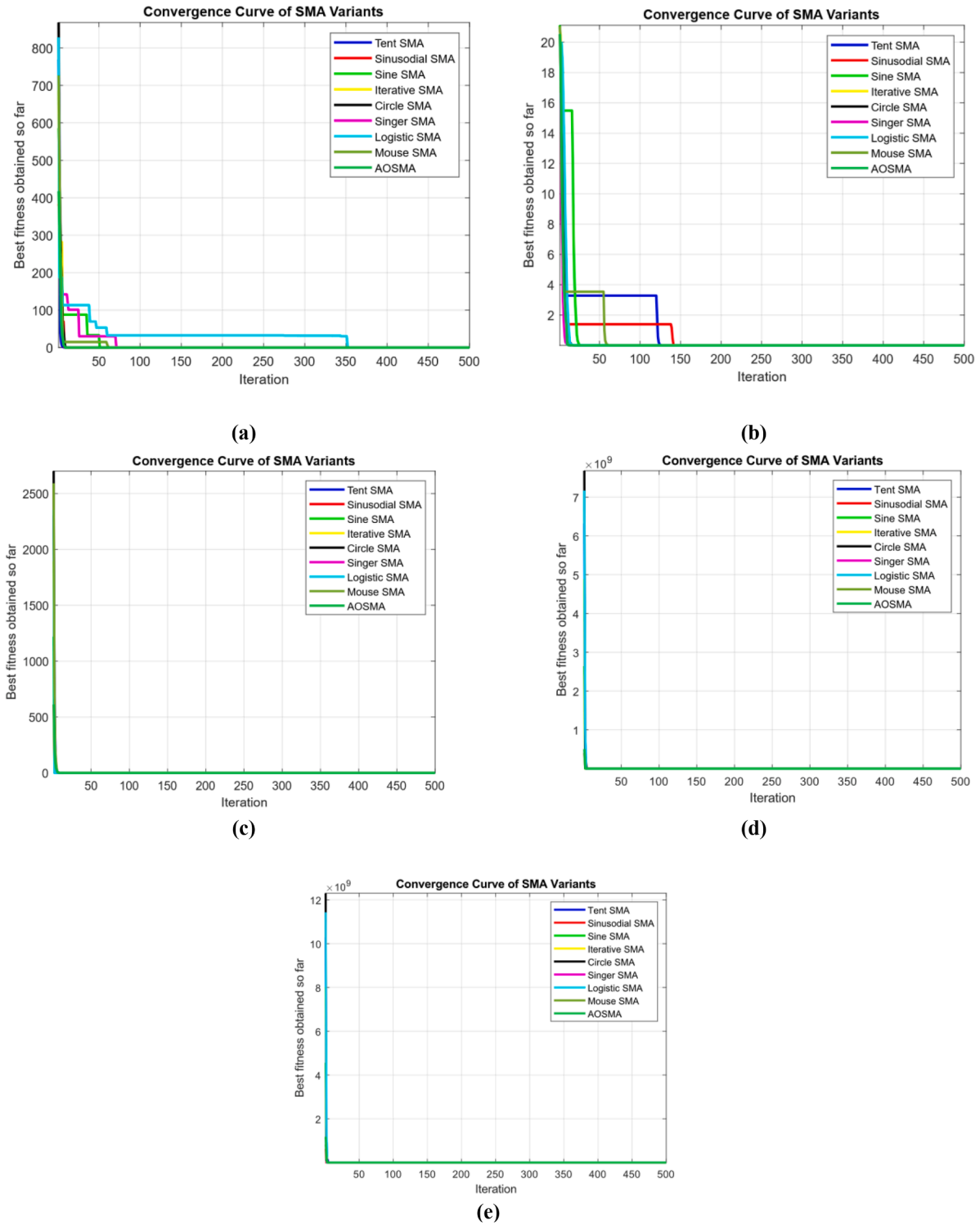

Fig. 3. (a) Convergence curve for f9, (b) Convergence curve for f10, (c) Convergence curve for f11, (d) Convergence curve for f12, (e) Convergence curve for f13 4 obtained by 9 SMA versions for benchmark functions (500 iterations).

$$c = 1 - \frac{t}{T} \tag{5}$$

where the maximum iteration is denoted by T. A fixed probability of 0.03 is assigned for randomly initializing a slime mould's position. The parameter p_i is a decision threshold for the i^{th} slime mould, which determines whether to update its position using the global best or retain its own position. It is calculated based on the fitness function f(x) as follows.

$$p_i = \tanh\left|f\left(X_i\right) - f_{GB}\right|, \ \forall\, i \in [1, N] \tag{6}$$

$$f_{GB} = f\left(X_{GB}\right) \tag{7}$$

where $f(x_i)$ is the fitness of the i^{th} agent X_i, and f_{GB} is the best global fitness. At the i^{th} iteration, the weight W for all N slime moulds is determined using the following equation.

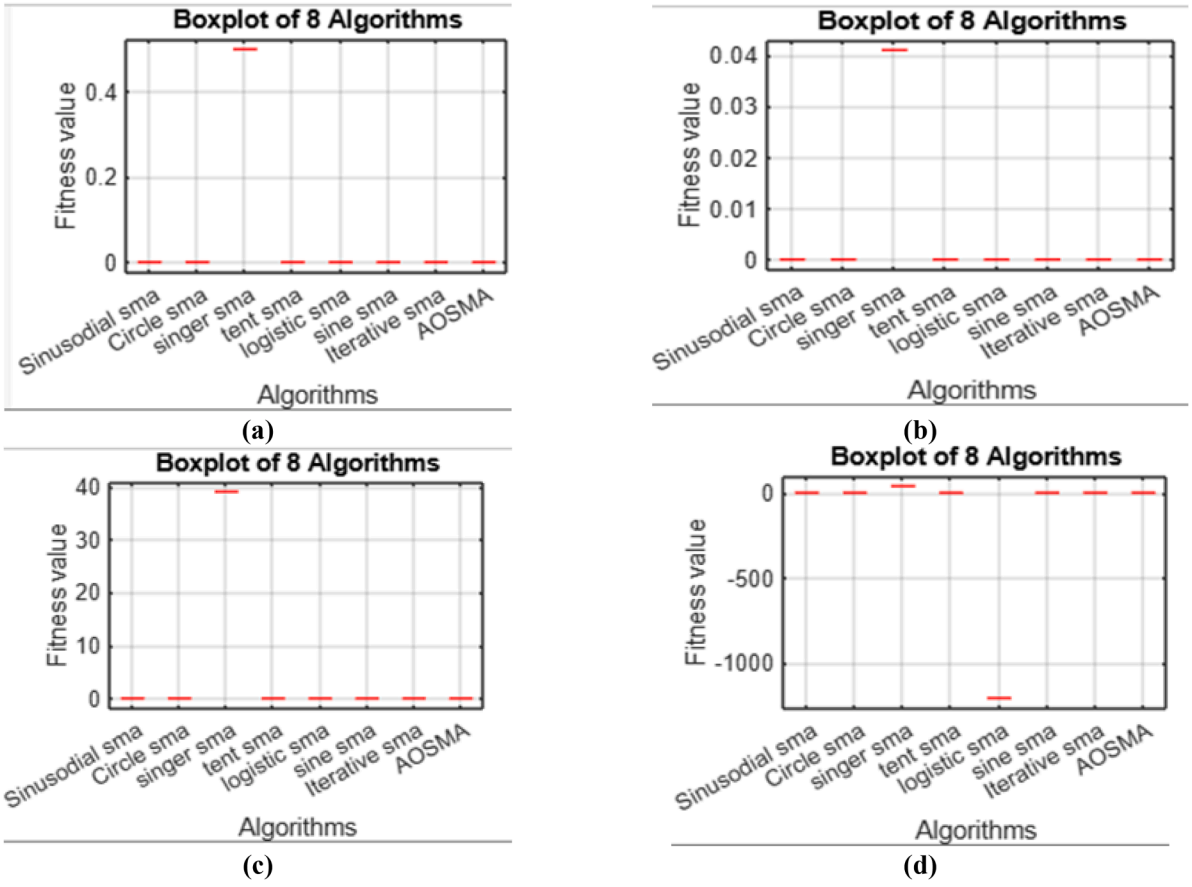

Fig. 4. (a) Boxplot for f1, (b) Boxplot for f2, (c) Boxplot for f3, (d) Boxplot for f4.

$$W(SortInd_f(i)) = \begin{cases} 1 + rand.\log\left(\frac{f_{LB} - f(X_i)}{f_{LB} - f_{LW}} + 1\right), & 1 \le i \le \frac{N}{2} \\ 1 + rand.\log\left(\frac{f_{LB} - f(X_i)}{f_{LB} - f_{LW}} + 1\right), & \frac{N}{2} < i \le N \end{cases}, \ \forall\, N \in \mathbb{R} \tag{8}$$

The terms f_{LB} and f_{LW} refer to the best and worst fitness values within the local population. For minimization tasks, the fitness values are sorted in increasing order.

$$[Sort_f,\ SortInd_f] = sort(f) \tag{9}$$

The optimal fitness value within the local region, along with the associated best-performing individual X_{LB}, is determined as follows.

$$f_{LB} = f(Sort_f(1)) \tag{10}$$

$$X_{LB} = X(SortInd_f(1)) \tag{11}$$

The local worst fitness f_{lw} is concluded as follows.

$$f_{LW} = f(Sort_f(N)) \tag{12}$$

3.2. Opposition-based learning

In OBL, for each slime mould (i = 1, 2, …, N), an opposite position X_{oi} is estimated relative to its current location X_{ni} in the search space. This value is compared to determine whether it offers a better solution for the next iteration, thus reducing the risk of getting trapped in local optima and enhancing convergence speed. The computation of X_{oi} in the j^{th} dimension is as follows.

$$X_{oij}(t) = \min(X_{ni}(t)) + \max(X_{ni}(t)) - X_{nij}(t) \tag{13}$$

3.3. Adaptive decision strategy

If the slime mould progresses along a path with decreasing nutrient quality, the algorithm adaptively responds using the current and previous fitness values within the AOSMA framework. Adaptive decision-making supports additional inquiry through OBL. AOSMA's adaptive decision approach is used to update the location for each subsequent iteration, as modeled below.

$$X_i(t+1) = \begin{cases} Xn_i(t) \ if\ f(Xn_i(t)) \le f(X_i(t)), \\ Xs_i(t) \ if\ f(Xn_i(t)) > f(X_i(t)), \end{cases} \quad \forall i \in [1, N] \tag{14}$$

$$Xs_i(t) = \begin{cases} Xo_i(t) \ if\ f(Xo_i(t)) < f(Xn_i(t)) \\ Xn_i(t) \ if\ f(Xo_i(t)) \ge f(Xn_i(t)) \end{cases} \tag{15}$$

4. Results and discussion

This section presents a comparative analysis between the proposed approach and other competing algorithms. All methods were tested under consistent conditions with a population size of 30 and 500 iterations, repeated five times. The proposed AOSMA algorithm was coded in MATLAB and executed on a machine with an Intel Core i7 2.20 GHz CPU and 12 GB RAM, using thirteen benchmark functions for evaluation.

The metrics that are utilized are Agent fitness values ($f(X_i)$), thresholds applied, and a decision threshold (p_i) that governs whether an agent maintains its current position or updates it toward the global best. To preserve diversity and prevent stagnation, a fixed probability of 0.03 is also used to reinitialize an agent's position randomly. Additionally, UB and LB vary for every function. Reassessment Frequency indicates that all agents' adaptive decisions are reassessed at each iteration. The

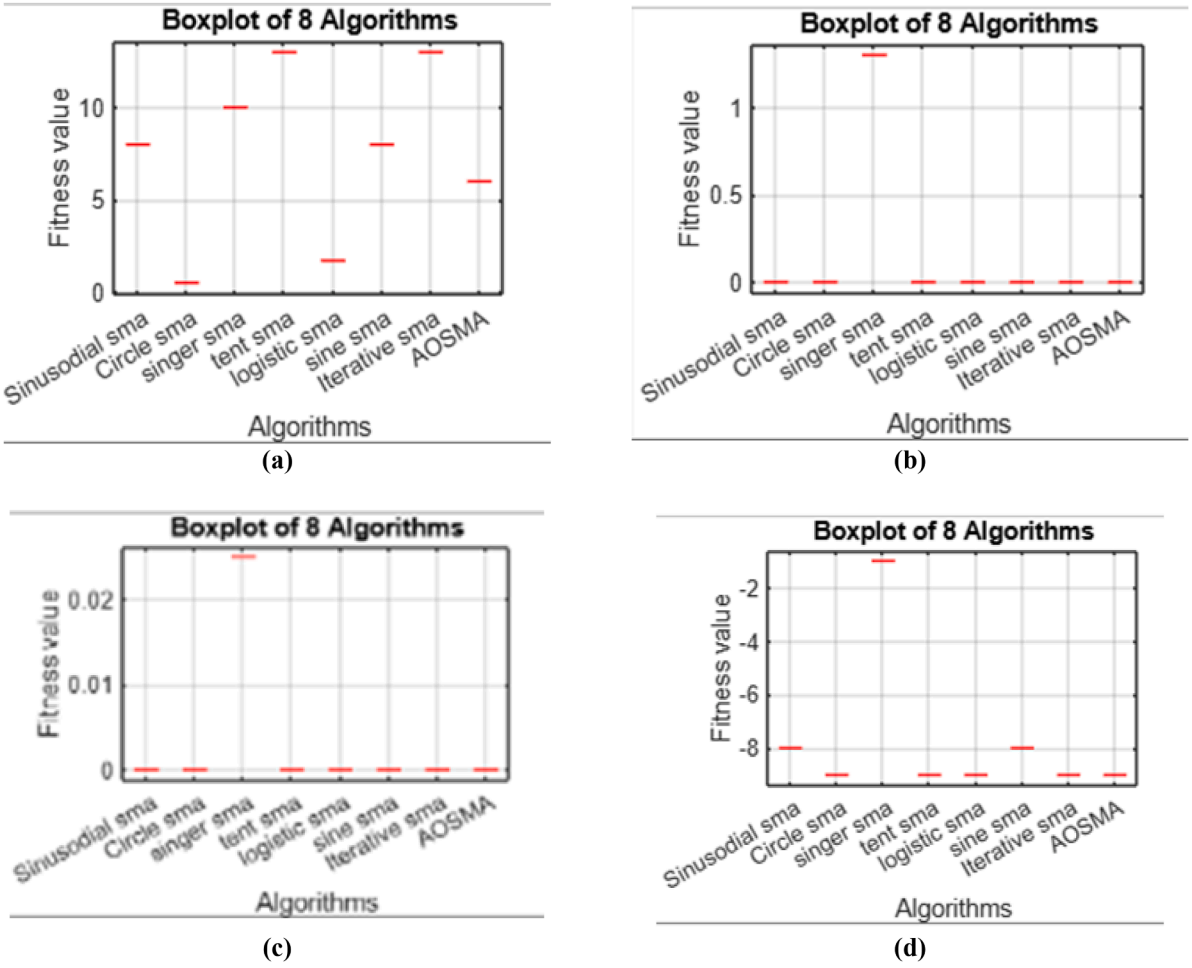

Fig. 5. (a) Boxplot for f5, (b) Boxplot for f6, (c) Boxplot for f7, (d) Boxplot for f8.

algorithm determines whether to initiate OBL or carry on with regular position updates at each iteration by comparing the current fitness to the prior fitness.

4.1. Simulation setup

Analysis of time and space complexity of an algorithm plays a central role in computational studies and practical applications, as it provides significant insights into their efficiency and scalability. Time complexity directly affects the algorithm's feasibility for large-scale or real-time problems. Similarly, space complexity quantifies the time of computation required in terms of memory, an essential aspect in discussing resource constraints, especially in embedded systems, big data systems, or limited-memory devices. Initially, the proposed AOSMA algorithm generates N search agents, each with a dimensionality of D, resulting in an initialization time complexity of O(N × D). The algorithm then evaluates the fitness of every agent throughout the optimization process, contributing to a total complexity of $O(t_{max}{\cdot}N{\cdot}D)$, where t_{max} is the maximum number of iterations. Furthermore, the AOSMA requires O (N·D) space, accounting for the storage of all search agents and their dimensions.

Experimental evaluations are conducted on thirteen conventional benchmarks. Table 3 summarizes mathematical formulations for the typical considered benchmarks. Unimodal benchmarks are commonly used to assess algorithms' exploitation capabilities due to their one global optimum. Basic functions are more difficult to manage than unimodal benchmarks due to their large number of local optima that can be used to measure exploration capabilities. Metaheuristics' ability to solve real-world engineering challenges can be estimated using common benchmarks. To ensure a fair comparison, 500 total iterations are used as the criterion condition, with a swarm size of 30. Eight methods are used to compare with AOSMA. To minimize the impact of chance on the experiment, all algorithms are conducted five times.

The selection of the 13 benchmark functions was guided by the need for a standardized, transparent, and reproducible framework to evaluate optimization algorithms across diverse problem characteristics. These benchmarks, including the widely used sphere function, were chosen for their prevalence in the optimization literature and the variety of mathematical properties they embody. Collectively, they cover separable and non-separable variables, unimodal and multimodal landscapes, and varying dimensionalities, providing a comprehensive test bed to assess the balance between exploration and exploitation in metaheuristic algorithms. Beyond their theoretical value, many of these functions capture features commonly encountered in real-world applications. For instance, the sphere function models convex and smooth landscapes typical of engineering design tasks requiring rapid convergence, while multimodal functions such as Rastrigin and Ackley emulate the complex, rugged search spaces found in scheduling, routing, and resource allocation problems.

F6 is not a special case of F7. $\sum_{i=1}^{n}(X_i+0.5)^2$ is not a special case of $\sum_{i=1}^{n}(X_i^4+rand\ (0,1))$ because the first expression is a deterministic mathematical operation that squares a value, while the second is a function that adds a random floating-point number to a constant. The expression $(X_i+0.5)^2$ performs addition and multiplication by 2, whereas $\sum_{i=1}^{n}(X_i^4+\ rand\ (0,1)$generates a random number between

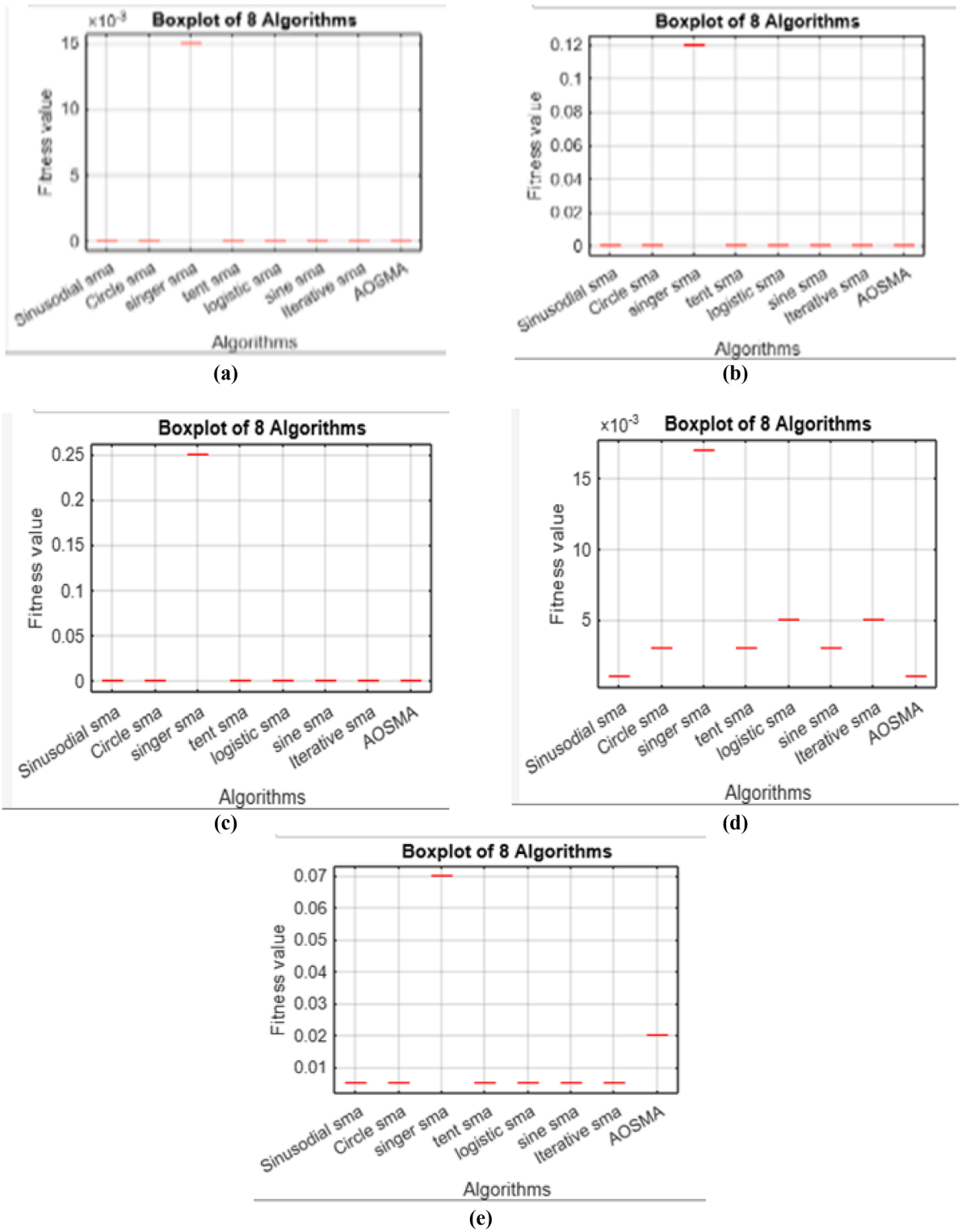

Fig. 6. (a) Boxplot for f9, (b) Boxplot for f10, (c) Boxplot for f11, (d) Boxplot for f12, (e) Boxplot for f13.

0 and 1 and adds it to the value of 1. Additionally, F7 cannot substitute F6 because they test different algorithmic properties. Both F6 (shifted sphere) and F7 (quartic with stochastic term) remain unchanged because they capture different features: simple convex convergence versus higher-order, noisy landscapes: F6 cannot replace F7 since it tests basic convergence behavior, whereas F7 analyzes resistance to noise and sensitivity to large variable magnitudes.

Parameters used in F6 can be explained as following, the shift parameter α=0.5 has been introduced in F6 in order to move the global minimum farther from the origin. Without the shift, the function simplifies to the conventional sphere model, whose symmetric and centered terrain makes it straightforward for many optimizers. The issue is transformed into a shifted-sphere function by setting α=0.5, which guarantees that optimizers cannot take advantage of origin symmetry and must instead show that they can find minima in a displaced search space. The selected value adds significant diversity to the benchmark

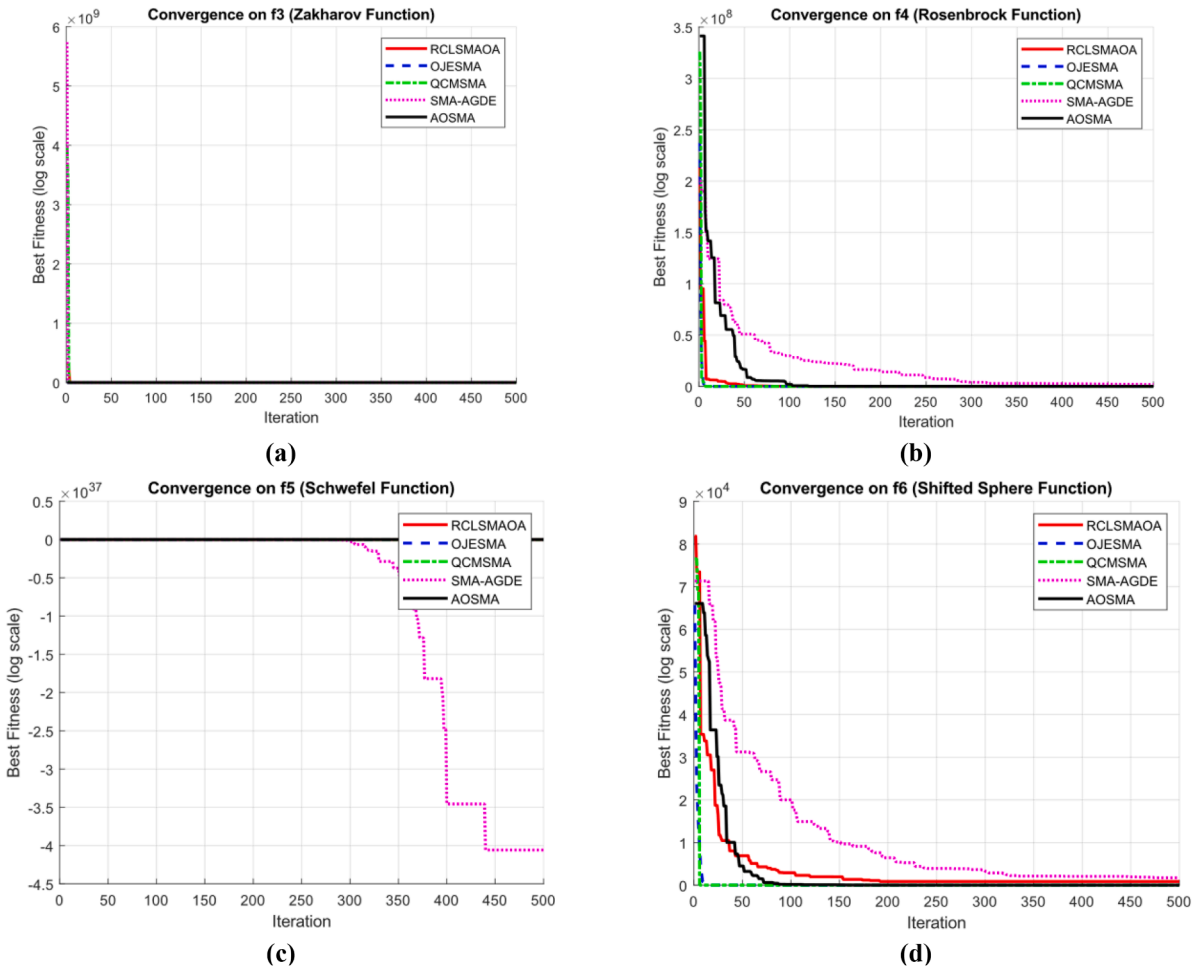

Fig. 7. Performance of the proposed AOSMA compared with the other four SMA versions [For F3-F6]. (a) Convergence curve for F3, (b) Convergence curve for F4, (c) Convergence curve for F5, (d) Convergence curve for F6, obtained by 5 SMA versions for benchmark functions (500 iterations).

suite while maintaining the function's convexity and simplicity. On the other hand, common shifts in benchmark functions include randomly generated shift vectors or fixed values like α=1. Random shifts offer greater diversity across runs, although larger shifts (e.g., α=1) increase the distance from the origin. In contrast to these, the reasonable selection of α=0.5 maintains the function's simplicity while making it non-trivial, balancing interpretability with difficulty. If α=0 it will be identical to F1.

To create tiny stochastic perturbations that mimic noise while maintaining the function's numerical stability and boundedness, the randomization in F7 is defined inside the interval (0,1). By making this decision, the random term is guaranteed to behave as a secondary disturbance rather than overpowering the function's predictable portion. The noise amplitude would significantly grow if the randomization were scaled to wider ranges, such (0100) or (0,1000). This might obscure the underlying landscape structure, skew the optimization difficulty, and possibly result in an unstable or deceptive performance evaluation. Because it strikes a compromise between realism and regulated complexity, the (0,1) interval is thus a common and suitable option.

4.2. Simulation results

This section compares the proposed AOSMA to the chaotic SMA on thirteen benchmark problems to show its effectiveness and competitiveness. Table 4 shows the experimental results for eight algorithms and the proposed AOSMA on thirteen typical benchmarks. AOSMA improves the performance of chaotic SMA on the unimodal tests F2, F3, and F4. Furthermore, the benefits of AOSMA over the chaotic SMA are retained. In most circumstances, AOSMA outperforms chaotic SMA in terms of competitiveness. AOSMA consistently achieves quicker convergence and greater ability to reach the global optimum compared to the chaotic SMA. The advantages of applying AOSMA in optimization are clearly presented in Table 4. It shows that AOSMA improves performance with 6 functions as F1, F2, F3, F4, F9, and F11. Additionally, this analysis shows its superior performance when compared to Mouse SMA, Iterative SMA, and the other remaining chaotic versions of SMA. While Mouse SMA, Iterative SMA, Sine SMA, Tent SMA, and Sinusoidal SMA improve performance on functions F1, F3, F9, and F11.

Figs. 1–3 present the convergence curve of the AOSMA and the other considered algorithms for the previously introduced thirteen benchmarks. Boxplots are utilized to assess the fitness performance of chaotic maps. The red line indicates the median fitness value of each algorithm. AOSMA shows the most favorable results with the lowest median, whereas Singer SMA records the highest. In Fig. 1(a), all SMA variants show a sharp decrease in fitness within the first 20–30 iterations, reflecting their strong exploratory behavior and their ability to identify promising areas of the search space rapidly. This rapid early improvement is characteristic of SMA-based algorithms, which naturally employ oscillatory movement patterns that enhance initial exploration. By approximately the 50th iteration, almost all methods, including the proposed AOSMA, achieve fitness values close to zero. This suggests that the benchmark problem is relatively straightforward for these algorithms, allowing them to converge quickly toward a near-optimal

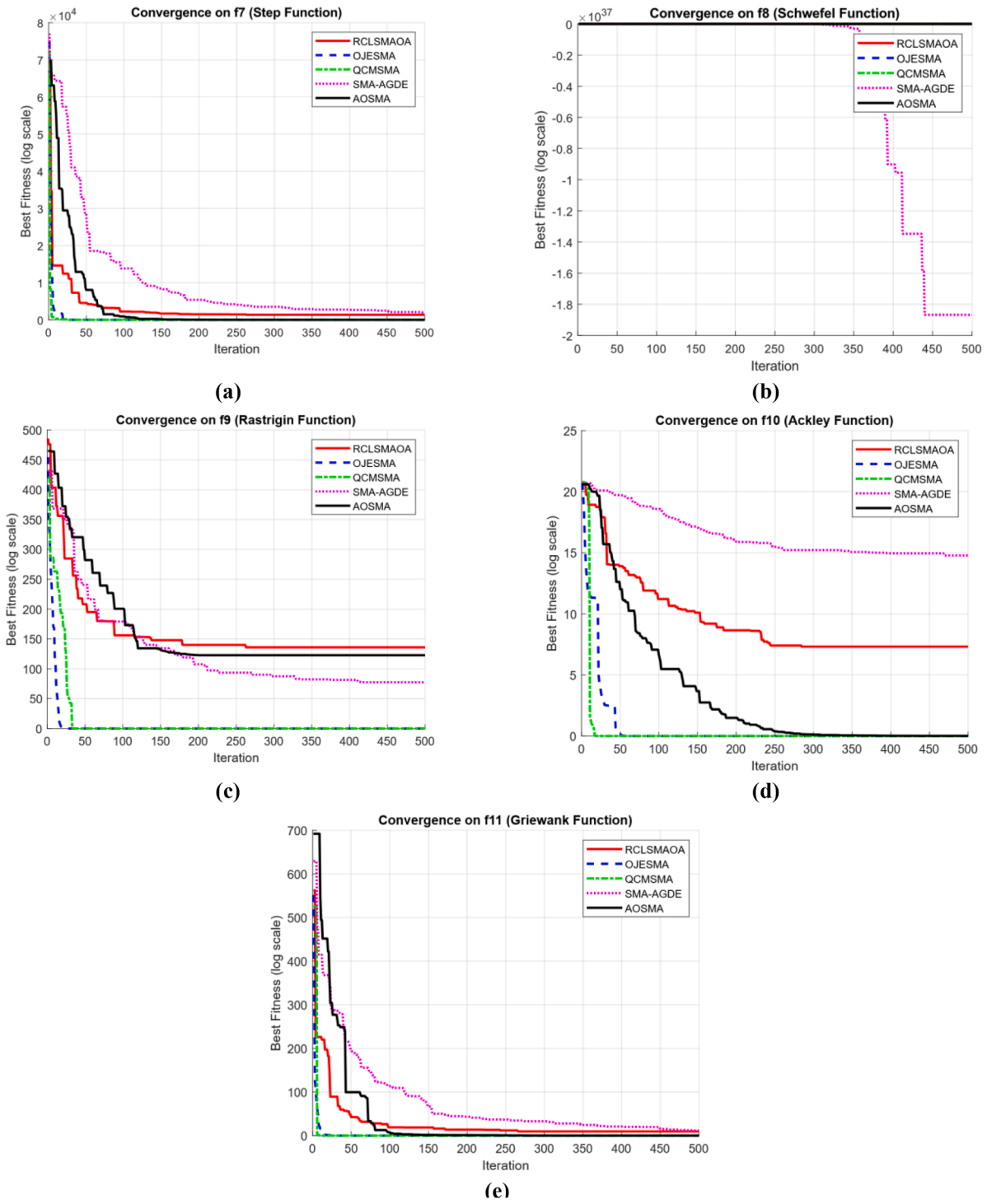

Fig. 8. Performance of the proposed AOSMA compared with the other four SMA versions [For F7-F11]. (a) Convergence curve for F7, (b) Convergence curve for F8, (c) Convergence curve for F9, (d) Convergence curve for F10, (e) Convergence curve for F11, obtained by 5 SMA versions for benchmark functions (500 iterations).

solution.

Overall, the figure shows that all SMA variants, including AOSMA, efficiently solve the benchmark problem, achieving fast convergence and almost identical final fitness values. AOSMA performs comparably to the top variants, confirming its effectiveness and indicating that this test function does not strongly distinguish performance differences among the SMA methods.

In Fig. 2(d), most SMA variants show a steep decline from high initial fitness values to significantly lower ones within the first 30–70 iterations, indicating that they quickly move away from inferior starting points and identify promising regions early in the search. Each curve corresponds to a specific SMA variant. Some variants achieve low fitness values rapidly and then level off, while others progress more gradually. Tent SMA, Sinusoidal SMA, and Sine SMA (red, yellow, and dark blue) exhibit rapid convergence followed by early stagnation, indicating intense short-term exploitation but limited long-term improvement. In contrast, Iterative SMA and Circle SMA exhibit similar convergence patterns but plateau at a later stage, reflecting a more moderate exploratory behavior.

Fig. 4-a presents the boxplot comparison of the fitness values derived

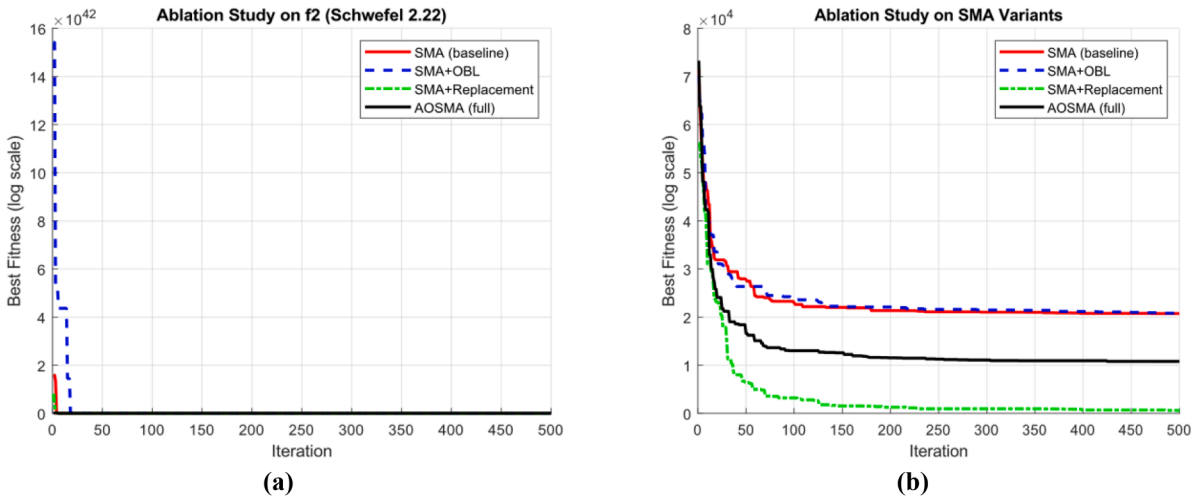

Fig. 9. Ablation study of four SMA versions at functions (a) F2 and (b) F1.

Table 5
Average wall-clock time per run and per iteration for AOSMA and SMA variants at sphere function F1, 30D, max iteration = 500.

Algorithm	Average runtime per run (s)	Average runtime per iteration (s)
Baseline SMA	0.4446	0.000889
AOSMA	0.6730	0.001346
Logistic SMA	0.4526	0.000905
Tent SMA	0.4576	0.000915
Sine SMA	0.6555	0.001311
Sinusoidal SMA	0.4628	0.000926
Singer SMA	0.4565	0.000913
Circle SMA	0.4530	0.000906
Mouse SMA	0.4481	0.000896
Iterative SMA	0.6288	0.001258

from eight distinct optimization techniques applied to the benchmark function F1. The y-axis shows the relevant fitness values, while the x-axis displays the algorithms: Sinusoidal SMA, Circle SMA, Singer SMA, Tent SMA, Logistic SMA, Sine SMA, Iterative SMA, and AOSMA. The graph clearly shows that the majority of algorithms perform well in optimization, showing very low or almost zero fitness values. In contrast to the others, the Circle SMA exhibits a noticeably higher spread and median value, indicating either worse performance or greater unpredictability. The AOSMA algorithm performs competitively and steadily among the compared approaches, as seen by its continually low fitness values.

Fig. 4-b presents a boxplot evaluating the effectiveness of eight optimization methods according to their fitness values using the benchmark F2. The algorithms, Sinusoidal SMA, Circle SMA, Singer SMA, Tent SMA, Logistic SMA, Sine SMA, Iterative SMA, and AOSMA, are represented on the x-axis, and the appropriate fitness values are displayed on the y-axis. The plot indicates effective optimization, as the majority of methods obtain very low fitness values. With a noticeably higher median and interquartile range, the Singer SMA stands out, indicating worse performance and more variability. Algorithms with minimum variance and lower fitness results, such as AOSMA and Tent SMA, on the other hand, exhibit excellent optimization capability and great stability.

Fig. 4-c depicts a boxplot of fitness values for eight optimization techniques, allowing a comparative examination of their performance for the F3 benchmark function. According to the statistics, Singer SMA generates a median fitness value that is significantly larger and has a broader range, which suggests less efficient optimization and greater performance variability. On the other hand, the remaining methods consistently obtain low fitness values, indicating higher optimization performance and robustness, especially AOSMA, Iterative SMA, and Tent SMA. This demonstrates that, out of the algorithms compared, AOSMA is among the most dependable and efficient. Fig. 4-d highlights the performance of eight optimization methods on the F4 benchmark function by comparing them in a boxplot according to their fitness values. With a noticeably lower (more negative) fitness value, the Logistic SMA sticks out from the plot, suggesting inadequate performance in this instance. The other methods, on the other hand, consistently maintain fitness values near zero, indicating superior optimization results. AOSMA and Iterative SMA are dependable options for addressing the optimization problem depicted because of their robust and constant performance.

Fig. 5 presents the fitness scores for the eight considered algorithms, for the benchmarks F5, F6, F7, and F8. Also, Fig. 6 presents the fitness scores for the benchmarks F9, F10, F11, F12, and F13. The algorithms are shown on the x-axis, and the fitness values, which reveal how well each algorithm optimizes, are displayed on the y-axis. The data shows that the algorithms differ noticeably from one another. With the lowest median fitness values, Circle SMA and Sine SMA appear to perform better in this situation. On the other hand, Singer SMA and AOSMA perform comparatively poorly on the examined task, as evidenced by their larger median fitness scores. The variations in the boxplots show how the algorithms differ in terms of robustness and stability.

For further evaluation, the proposed AOSMA was compared to four other SMA versions. Figs. 7 and 8 present the results of the proposed AOSMA compared to the other four SMA versions for benchmark functions of F3-F11. Every algorithm exhibits a distinct decreasing trend, indicating that as the number of iterations increases, the quality of the solutions improves. The magenta (SMA-AGDE) and green (QCMSMA) curves exhibit remarkably rapid convergence, achieving very low fitness values early on and maintaining consistent gains throughout. Throughout, the blue (QJESMA) curve shows a steady decline that smoothly converges to lower fitness values. As iterations progress, the black (AOSMA) and red (RCLSMOA) curves exhibit stable convergence behavior, rapidly declining from their initial higher values before stabilizing. The convergence curves of five algorithms over 500 iterations on the Rosenbrock function (f4) are displayed in this picture.

Every algorithm begins with extremely high fitness values, ranging from 10 to 8, and drops off quickly in the initial repetitions. Within the first 50 iterations, RCLSMAOA, OJESMA, and QCMSMA show rapid convergence, achieving values that are nearly zero. SMA-AGDE improves progressively over iterations and converges more slowly. AOSMA exhibits competitive performance, with a smooth transition to near-optimal values and a distinct, steady decline. The Rosenbrock function is handled well by all approaches overall, albeit there are variations in

stability and convergence rate. OJESMA, QCMSMA, and RCLSMAOA all perform extremely steadily and swiftly approach near-optimal values. SMA-AGDE has a greater fitness trajectory over the course of the iterations and converges significantly more slowly. AOSMA exhibits a balance: it performs competitively and converges more smoothly than SMA-AGDE; however, it lags somewhat behind RCLSMAOA and QCMSMA in terms of final precision.

AOSMA's parameters were selected to maintain a balanced search behavior while ensuring practical implementation and reproducibility, using standard SMA defaults to minimize manual tuning. Uniform settings for population size, iteration count, and reinitialization probability produced stable performance across benchmarks. Although initial findings indicate that excessively large populations or frequent OBL activations offer limited additional benefit due to the algorithm's inherent diversity mechanisms, further sensitivity studies, such as exploring different population sizes or OBL activation strategies, would provide more quantitative insight and help confirm the robustness of these parameter choices. Incorporating real-world optimization case studies would further strengthen the manuscript's practical impact, as benchmark functions do not fully capture the constraints, noise, and interdependencies present in actual applications. Since AOSMA's adaptive opposition learning and best-agent replacement strategies are particularly effective for complex and resource-constrained problems, evaluating the algorithm on tasks such as engineering design or feature selection would illustrate its practical performance and expand its applicability without altering its core framework.

AOSMA exhibits consistent performance as dimensionality grows, largely due to its adaptive opposition mechanism, which preserves population diversity, and the best-agent replacement strategy, which promotes faster convergence toward promising regions. In 30-dimensional benchmark evaluations, the algorithm consistently achieved competitive or superior outcomes across unimodal, multimodal, and composite problem classes. Its complexity scales linearly with dimension, $O(ND)$ per iteration and $O(T_{\max}ND)$overall, preventing exponential increases in runtime or degradation in solution quality. Consequently, AOSMA maintains robust search efficiency in higher-dimensional spaces without requiring specialized parameter tuning. Future work may extend these findings to ultra-high-dimensional settings (e.g., $D > 100$) or real-world feature selection tasks to further assess scalability.

Furthermore, Fig. 9 presents an ablation study of four SMA versions at functions F1 and F2. The red curve represents the SMA baseline. The standard SMA begins with high fitness values and decreases gradually, but plateaus at a higher level than other variants, reflecting weaker optimization ability. The blue dashed curve represents the SMA + OBL. Incorporating non-adaptive OBL yields only a slight improvement over the baseline, with a convergence trend that nearly overlaps it, indicating limited effectiveness. The green dash-dotted curve represents the SMA + replacement. This variant demonstrates the fastest and most pronounced improvement, quickly reaching lower fitness values and maintaining strong convergence, highlighting the effectiveness of the replacement mechanism. The black curve represents the proposed AOSMA. By integrating adaptive OBL with replacement, AOSMA achieves stable and competitive performance. Although its early progress is slower than SMA+ Replacement, it ultimately converges to the lowest fitness, offering the best balance between exploration and exploitation.

A non-parametric statistical test for comparing the performance of three or more algorithms (or treatments) across various problems or datasets is the Friedman Test, often known as Friedman's ANOVA. In optimization and machine learning research, it is very typical to seek to determine whether one algorithm (like AOSMA) performs noticeably better than others. Here's what the Friedman analysis (F1–F9 subset) shows:

a) Average ranks: AOSMA achieves the lowest average rank (best overall), indicating it tends to perform more strongly across functions.

b) Friedman test statistic: Since the p-value is 0.0002 (< 0.05), the differences among algorithms are statistically significant. This means we can reject the null hypothesis ("all algorithms are equal") and conclude that at least one algorithm, in this case, AOSMA, with the best average rank, is significantly better.

Table 5 presents the average wall-clock runtime per run and per iteration for the baseline SMA, its chaotic variants, and AOSMA on the Sphere benchmark function (30 dimensions, 30 agents, 500 iterations). The baseline SMA and most chaotic variants (Logistic, Tent, Sinusoidal, Singer, Circle, Mouse, Iterative) demonstrate relatively low runtimes, averaging around 0.44–0.46 s per run, with per-iteration costs on the order of 10^{-3} seconds. Within the chaotic variants, Sine SMA and Iterative SMA show slightly higher runtimes, reflecting the extra computational effort introduced by their nonlinear update dynamics.

In contrast, AOSMA records a moderately higher runtime of 0.673 s per run (0.001346 s per iteration). This increase is anticipated since AOSMA incorporates opposition-based learning and adaptive exploitation mechanisms, which add extra computational steps per iteration. Nonetheless, this modest selection is offset by AOSMA's superior optimization accuracy, stability, and convergence speed, as evidenced by the experimental results. Ultimately, the added computational cost represents a reasonable trade-off for achieving improved search balance and higher solution quality, reaffirming the effectiveness of AOSMA in enhancing slime mould-based optimization.

OBL increases computational overhead by effectively doubling the number of candidate solutions, both the original and their opposites, that must be evaluated and maintained. Similarly, replacement strategies, which determine which individuals advance to the next generation, introduce additional costs through comparison and selection operations. Although these mechanisms enhance optimization by boosting population diversity and reducing the risk of premature convergence, the extra computations required for generating and filtering solutions contribute to the overall runtime. As a result, they may be less practical for highly cost-sensitive optimization tasks or scenarios with limited computational resources.

Two additional mechanisms-opposition-based learning, which assesses opposing solutions, and the replacement strategy, which swaps out weaker agents for stronger ones- are responsible for the extra runtime in AOSMA. Compared to the entire optimization process, these phases are lightweight and add a little computational burden that scales linearly with issue size. Importantly, the overhead is a good trade-off because this minor expense results in substantial gains in exploration, exploitation, and solution quality.

5. Conclusion and future work

This study proposed an AOSMA aimed at addressing function optimization challenges. However, because the suggested algorithm makes an adaptive decision about whether to use the OBL, it shows superior exploration and exploitation. In conclusion, this approach reduces the possibility of misguiding the exploration phase in some cases by 50 % by using a single random search agent instead of the SMA. Additionally, this method has an adaptive mechanism built in to determine when to utilize OBL to limit the exploration period. Lastly, the proposed technique is thought to be helpful for function optimization to address practical engineering issues.

As a future work, we can combine with metaheuristics (e.g., PSO, DE, GA) to leverage strengths and overcome weaknesses (e.g., slow convergence or premature convergence). Additionally, including one or two state-of-the-art optimizers outside the SMA family, consider using larger benchmark sets or higher dimensionalities would better evaluate the proposed method's generalization and robustness. Also, this would offer deeper insights into the effectiveness of the proposed approach.

Data availability statement

The data that support the findings of this study are available in the manuscript. Further information related to the considered data can be obtained by reasonable request from the corresponding author.

Funding

This work was funded by Prince Sultan University.

CRediT authorship contribution statement

Elsayed Badr: Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Mostafa Abdullah Ibrahim:** Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Diaa Salama:** Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Mohammed ElAffendi:** Writing – review & editing, Visualization, Validation, Supervision, Software, Resources, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Abdelhamied A Ateya:** Writing – review & editing, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Mohamed Hammad:** Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Alaa Yassin:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The Authors would like to thank Prince Sultan University for paying the APC of this article.

References

[1] M. Zhou, M. Cui, D. Xu, S. Zhu, Z. Zhao, A. Abusorrah, Evolutionary optimization methods for high-dimensional expensive problems: a survey, *IEEE/CAA J. Autom. Sin.* 11 (5) (2024) 1092–1105.
[2] E.H.H. Sumiea, et al., Enhanced deep deterministic policy gradient algorithm using grey wolf optimizer for continuous control tasks, IEEE Access 11 (2023) 139771–139784.
[3] E.H. Houssein, M.K. Saeed, G. Hu, M.M. Al-Sayed, Metaheuristics for solving global and engineering optimization problems: review, applications, open issues and challenges, Arch. Comput. Methods Eng. 31 (8) (2024) 4485–4519.
[4] R. Rishabh, K.N. Das, A critical review on metaheuristic algorithms based multi-criteria decision-making approaches and applications, Arch. Comput. Methods Eng. 32 (2) (2024) 963–993.
[5] F.S. Gharehchopogh, A. Ucan, T. Ibrikci, B. Arasteh, G. Isik, Slime Mould algorithm: a comprehensive survey of its variants and applications, Arch. Comput. Methods Eng. 30 (4) (2023) 2683–2723.
[6] T. Yu, J. Pan, Q. Qian, Improved slime mould algorithm by perfecting bionic-based mechanisms, Int. J. Bio-Inspired Comput. 22 (1) (2023) 1–15.
[7] H. Chen, Z. Wang, H. Jia, X. Zhou, L. Abualigah, Hybrid slime mold and arithmetic optimization algorithm with random center learning and restart mutation, Biomimetics 8 (5) (2023) 396 *(Basel)*.
[8] S. Larabi-Marie-Sainte, R. Alskireen, S. Alhalawani, Emerging applications of bio-inspired algorithms in image segmentation, Electronics 10 (24) (2021) 3116 *(Basel)*.
[9] I. Abunadi, et al., An automated glowworm swarm optimization with an inception-based deep convolutional neural network for COVID-19 diagnosis and classification, Healthcare 10 (4) (2022) 697 *(Basel)*.
[10] M.K. Naik, R. Panda, A. Abraham, Adaptive opposition slime mould algorithm, Soft Comput. 25 (22) (2021) 14297–14313.
[11] J. Yang, Y. Zhang, T. Jin, Z. Lei, Y. Todo, S. Gao, Maximum Lyapunov exponent-based multiple chaotic slime mold algorithm for real-world optimization, Sci. Rep. 13 (1) (2023) 12744.
[12] H. Chen, C. Li, M. Mafarja, A.A. Heidari, Y. Chen, Z. Cai, Slime mould algorithm: a comprehensive review of recent variants and applications, Int. J. Syst. Sci. 54 (1) (2023) 204–235.
[13] W.-C. Wang, W.-H. Tao, W.-C. Tian, H.-F. Zang, A multi-strategy slime mould algorithm for solving global optimization and engineering optimization problems, Evol. Intell. 17 (5–6) (2024) 3865–3889.
[14] Z. Duan, X. Qian, W. Song, Multi-strategy enhancde slime mould algorithm for optimization problems, IEEE Access 13 (2025) 1. –1.
[15] E.H. Houssein, M.A. Mahdy, M.J. Blondin, D. Shebl, W.M. Mohamed, Hybrid slime mould algorithm with adaptive guided differential evolution algorithm for combinatorial and global optimization problems, Expert Syst. Appl. 174 (114689) (2021) 114689.
[16] P.V.H. Son, L.N.Q. Khoi, Optimization in construction management using adaptive opposition slime mould algorithm, Adv. Civ. Eng. 2023 (2023) 1–20.
[17] W. Xiong, D. Li, D. Zhu, R. Li, Z. Lin, An enhanced slime mould algorithm combines multiple strategies, Axioms 12 (10) (2023) 907.
[18] Laith Abualigah, Ali Diabat, Mohamed Abd Elaziz, Improved slime mould algorithm by opposition-based learning and Levy flight distribution for global optimization and advances in real-world engineering problems, J. Ambient. Intell. Humaniz. Comput. 14 (2) (2023) 1163–1202.