Received: July 30, 2025. Revised: November 7, 2025. 221

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Bolas Spider Algorithm: A Novel Efficient Nature-Inspired Metaheuristic for Complex Continuous Optimization

Haitham Qawaqneh[1] Safwan Maghaydah[2] Saleh Alomari[3] Gulnara Bektemyssova[4] Zeinab Montazeri[5] Mohammad Dehghani[5]* Om Parkash Malik[6] Kei Eguchi[7]

[1]*Department of Mathematics, Al Zaytoonah University of Jordan, Amman 11733, Jordan*
[2]*Faculty of Information Technology, Abu Dhabi University, UAE*
[3]*Faculty of Science and Information Technology, Software Engineering, Jadara University, Irbid 21110, Jordan*
[4]*Department of Computer Engineering, International Information Technology University, Almaty 050000, Kazakhstan*
[5]*Department of Electrical and Electronics Engineering, Shiraz University of Technology, Shiraz 7155713876, Iran*
[6]*Department of Electrical and Software Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada*
[7]*Department of Information Electronics, Fukuoka Institute of Technology, Japan*
* Corresponding author's Email: adanbax@gmail.com

Abstract: A novel metaheuristic optimization algorithm named Bolas Spider Algorithm (BSA), inspired by the hunting behaviour of bolas spiders is presented in this paper. The proposed algorithm combines pheromone-guided exploration with targeted prey-capture exploitation to achieve a dynamic balance between diversification and intensification, enabling effective navigation of complex continuous optimization landscapes. The algorithm's design emphasizes adaptability, robustness, and high solution quality, while avoiding premature convergence and maintaining population diversity. Performance of the algorithm was rigorously evaluated on a comprehensive benchmark suite comprising 29 continuous functions, including unimodal, multimodal, hybrid, and composition problems. Comparative experiments involved nine recently developed metaheuristic algorithms, and multiple statistical measures—mean, best, worst, standard deviation, median, and rank—were computed over 30 independent runs for each function. Additionally, the Wilcoxon signed-rank test has been employed to validate the statistical significance of the results. Empirical findings indicate that the proposed algorithm consistently achieves superior performance, obtaining the first rank in 24 out of 29 benchmark functions, including all composition functions and several complex multimodal and hybrid problems. Qualitative analysis using boxplot visualizations further confirms the algorithm's stability and robustness, demonstrating narrow distributions, low variability, and minimal outliers across independent runs. The observed advantages are attributed to the algorithm's dual search mechanism, which efficiently combines global exploration with local exploitation, ensuring both convergence accuracy and repeatability. Overall, the results establish the proposed algorithm as a highly effective, reliable, and statistically validated optimization method. Its biologically inspired mechanisms and parameter-efficient design make it suitable for a wide range of continuous optimization problems, with potential applications in engineering, industrial, and real-world decision-making scenarios.
Keywords: Bolas spider algorithm, Metaheuristic optimization, Continuous optimization, Exploration and Exploitation, Benchmark functions, Algorithm robustness.

1. Introduction

Optimization represents a cornerstone of computational science, mathematics, and engineering, dealing with the process of identifying the most effective solution among a set of feasible alternatives

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

according to one or more objective functions. In essence, an optimization problem seeks to minimize or maximize a particular objective function under a given set of constraints that define the solution space. Such problems arise across an extensive range of real-world domains, including structural design, power system management, transportation, data clustering, and machine learning. The goal of optimization is to achieve the best possible outcome—whether minimizing cost and energy consumption or maximizing performance, accuracy, or efficiency. However, the complexity of these problems increases rapidly with dimensionality, nonlinearity, and the presence of local optima, making conventional analytical methods insufficient for solving them effectively [1-5].

Generally, optimization techniques are classified into two broad categories: deterministic and stochastic approaches [6]. Deterministic optimization algorithms rely on explicit mathematical models and well-defined rules to move systematically toward the optimal solution. Examples include linear programming, quadratic programming, gradient descent, and Newton-based methods. These algorithms are often efficient for convex and differentiable functions, where the objective function and constraints are continuous and well-behaved [7, 8]. Their major advantage lies in their mathematical rigor and ability to guarantee global optimality under certain conditions. However, deterministic techniques also face significant limitations. They often require gradient information, are sensitive to the initial point, and tend to converge to local optima when dealing with multimodal or nonconvex problems [9, 10]. Moreover, deterministic methods typically perform poorly when the objective function is discontinuous, noisy, or highly nonlinear, as they lack the flexibility to explore complex landscapes effectively [11].

To address these drawbacks, stochastic or probabilistic optimization techniques, that incorporate randomness and probabilistic decision rules to escape local optima and explore the search space more effectively, have been introduced. These approaches are particularly suitable for large-scale, nonconvex, and NP-hard problems where deterministic methods are impractical or computationally expensive. Within this stochastic family, metaheuristic algorithms have emerged as a dominant paradigm due to their flexibility, simplicity, and high adaptability to various problem types [12]. Metaheuristics are high-level frameworks that guide subordinate heuristics to explore solution spaces efficiently, often inspired by natural, biological, social, or physical phenomena [13]. Their primary goal is to balance two conflicting processes in optimization: exploration, which ensures diverse sampling across the search space to discover new promising regions, and exploitation, which intensifies the search around known good solutions to improve accuracy [14].

The fundamental advantage of metaheuristic algorithms lies in their problem-independent design, requiring minimal mathematical information about the optimization problem. Unlike deterministic techniques, they do not rely on derivatives or convexity assumptions. Instead, they use simple mathematical operators—such as random walks, velocity updates, or adaptive parameter tuning—to iteratively improve candidate solutions. A typical metaheuristic begins with a population of randomly generated solutions representing different points in the search space. During each iteration, these solutions are evaluated according to a fitness function, and their positions are updated through a series of rules that model natural processes such as reproduction, migration, communication, or physical attraction. Over time, the population collectively converges towards the global optimum or an approximation of it. Although convergence to the true global optimum cannot be theoretically guaranteed, metaheuristics provide highly competitive solutions within reasonable computational time, making them indispensable for real-world applications [15].

A central concept in metaheuristic optimization is the dynamic balance between exploration and exploitation. Exploration refers to the algorithm's ability to diversify the search and avoid premature convergence by investigating new, unvisited regions of the solution space. In contrast, exploitation focuses on refining the best-known solutions by intensifying the search in their local neighborhood [16]. An effective metaheuristic algorithm must maintain an adaptive equilibrium between these two components. Excessive exploration can waste computational effort and slow convergence, while excessive exploitation can cause stagnation in suboptimal regions. Various strategies—such as adaptive parameter tuning, hybrid search mechanisms, and feedback-based learning—have been developed to achieve this balance dynamically during the optimization process [17].

Despite their advantages, metaheuristic algorithms have an inherent limitation: they are non-deterministic and provide no formal guarantee of reaching the true global optimum. Their stochastic nature implies that outcomes may vary between runs, and convergence is often problem-dependent [18]. Nonetheless, the flexibility and robustness of these algorithms have led to widespread adoption in

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

diverse fields ranging from engineering design and scheduling to bioinformatics and artificial intelligence. Over the past three decades, the continuous quest for improved performance has driven researchers to develop numerous metaheuristic algorithms inspired by nature, physics, and human behavior. These algorithms differ in their inspiration sources and search mechanisms but share a common structural framework centered on population-based search and iterative refinement [11].

Among the most influential and widely adopted algorithms, the Particle Swarm Optimization (PSO) [19] algorithm stands as a seminal example of swarm intelligence. PSO mimics the collective behavior of bird flocks or fish schools, where each particle represents a potential solution navigating through the search space by adjusting its velocity according to its own best-known position and that of its neighbors. The algorithm's simplicity, efficiency, and strong global search capabilities have made it one of the most popular metaheuristics for both continuous and discrete optimization problems. Similarly, the Genetic Algorithm (GA) [20], inspired by the process of natural selection, employs mechanisms such as selection, crossover, and mutation to evolve a population of candidate solutions toward optimality. GA's evolutionary structure enables it to effectively explore large and complex search spaces, though it may sometimes struggle with convergence speed and parameter tuning.

The Ant Colony Optimization (ACO) [21] algorithm, another pioneering method, draws inspiration from the foraging behavior of ants, which deposit pheromones to communicate promising paths to food sources. By translating this biological process into a computational framework, ACO has proven effective for combinatorial optimization problems such as the traveling salesman problem and scheduling. The Simulated Annealing (SA) [22] algorithm, inspired by the annealing process in metallurgy, provides a probabilistic framework for escaping local optima by occasionally accepting worse solutions early in the search, gradually reducing this probability as the temperature parameter decreases. SA's strength lies in its theoretical simplicity and ability to avoid premature convergence, though it can be computationally slow for large-scale problems.

The Grey Wolf Optimizer (GWO) [23], a more recent nature-inspired approach, models the social hierarchy and cooperative hunting behavior of grey wolves. GWO introduces a leadership-based hierarchy, including alpha, beta, delta, and omega wolves, to simulate coordinated pursuit and encircling of prey, offering a well-balanced trade-off between exploration and exploitation. Teaching-Learning-Based Optimization (TLBO) [24], inspired by the interaction between teachers and students in a classroom, eliminates the need for algorithm-specific parameters. It operates through two phases: the "teaching" phase, where learners improve their knowledge guided by the teacher (best solution), and the "learning" phase, where learners exchange information with one another. TLBO's parameter-free design and strong convergence characteristics have made it appealing for various engineering applications. The Gravitational Search Algorithm (GSA) [25] relies on the Newtonian law of gravity and motion, treating agents as objects with masses that attract one another proportionally to their fitness. The combination of gravitational force and motion equations enables agents to move toward global optima with adaptive exploration and exploitation behavior. Collectively, these classical algorithms have established the conceptual and methodological foundation upon which newer metaheuristics continue to build, each attempting to overcome specific weaknesses such as premature convergence, parameter dependency, and limited adaptability.

In recent years, an extensive wave of innovation has led to the introduction of numerous new metaheuristic algorithms inspired by human behavior, biological systems, physical processes, and hybrid mathematical models. One of the recent human-inspired methods is the Mother Optimization Algorithm (MOA) [26], which models the nurturing, guiding, and disciplinary behaviors of a mother toward her children. MOA simulates three main phases—nurturing, guidance, and discipline—to dynamically control the search process. The nurturing phase corresponds to exploration, fostering solution diversity; the guidance phase refines candidate solutions based on the best performers; and the discipline phase eliminates poor solutions, reinforcing exploitation. This structured behavioral hierarchy enhances convergence precision and robustness while maintaining balance between global and local search.

Another notable innovation is the Adams-Bashforth-Moulton Optimizer (ABMO) [27], inspired by the numerical integration method combining explicit and implicit schemes in differential equation solving. ABMO employs a two-step prediction-correction mechanism in which the prediction stage enhances exploration by estimating potential promising regions, while the correction stage exploits local information to refine candidate solutions. This dual-step process ensures stability, accuracy, and efficient convergence, offering significant improvements over traditional

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

population-based algorithms that rely solely on random updates.

The Traffic Jam Optimizer (TJO) [28] introduces an entirely different metaphor based on traffic regulation dynamics. It simulates the behavior of vehicles and traffic police during congestion and clearance phases. Initially, autonomous driving leads to disordered movement (exploration), followed by a self-regulation stage where drivers adjust directions based on local information, and finally, an enforcement phase where a traffic controller guides the system toward smooth flow (exploitation). This multi-phase mechanism provides an adaptive balance between randomness and control, effectively preventing stagnation in local optima.

Another promising approach is the Octopus Optimization Algorithm (OOA) [29], inspired by the intelligent and flexible movement of octopuses. The algorithm models both the exploratory locomotion and exploitative hunting strategies of octopuses to navigate complex search spaces. Its multi-objective extension (MOOA) employs population grouping and dominance ranking to manage conflicting objectives, maintaining solution diversity and ensuring convergence toward Pareto-optimal fronts. The Octopus framework exemplifies how biologically grounded intelligence can be formulated mathematically to handle both single and multi-objective optimization challenges.

The Artificial Lemming Algorithm (ALA) [30] draws inspiration from the behavioral ecology of lemmings, small mammals known for their collective migration and survival strategies. ALA incorporates four behaviors—migration, burrowing, foraging, and predator avoidance—each representing distinct phases of exploration and exploitation. The algorithm also integrates an energy-decreasing mechanism that dynamically adjusts exploration intensity as the search progresses. This adaptive control enhances robustness, making ALA effective in escaping local traps and solving high-dimensional nonconvex problems.

In contrast to population-based algorithms, the Single Candidate Optimizer (SCO) [31] represents a minimalistic yet efficient approach that operates using only a single solution throughout the entire search process. By applying distinct update rules in its two-phase mechanism, SCO alternates between broad exploration and refined exploitation. Despite its simplicity, empirical studies have shown that SCO can outperform many population-based algorithms, demonstrating that efficient search dynamics can be achieved without reliance on large populations or complex parameters.

The Crested Porcupine Optimizer (CPO) [32] is another bio-inspired algorithm, modeled after the layered defensive behaviors of the crested porcupine, which employs sight, sound, odor, and physical attack as sequential defense strategies. CPO categorizes these behaviors into exploration and exploitation phases, where early defensive strategies (sight and sound) drive global search, and later strategies (odor and attack) intensify local search. The algorithm introduces a cyclic population reduction strategy to mimic natural selective activation, improving convergence rate and preserving diversity within the population.

The Snake Optimizer (SO) [33] simulates the distinctive foraging and mating behavior of snakes under environmental constraints such as temperature and food availability. The algorithm distinguishes between male and female agents, each adopting different movement strategies depending on environmental conditions. When resources are abundant and temperature is favorable, snakes compete for mates, representing an exploitative process; under scarcity, they disperse, enhancing exploration. The dynamic switching between these behavioral states provides SO with adaptive search capabilities suited for complex, multimodal problems.

The Artificial Rabbits Optimization (ARO) [34] algorithm is inspired by the survival and foraging behavior of rabbits, particularly their detour foraging and random hiding strategies. These mechanisms are mathematically modeled to alternate between exploration and exploitation phases. During the detour foraging stage, rabbits explore the search domain by feeding near the nests of others, thereby promoting diversity and preventing premature convergence. As energy decreases, rabbits switch to random hiding, concentrating the search around promising regions. This biologically inspired adaptability grants ARO high robustness and convergence efficiency.

The Rabbit and Turtle Algorithm (RTA) [35] presents a dual-agent framework inspired by the contrasting behaviors of a rabbit and a turtle. The rabbit symbolizes rapid, stochastic exploration, while the turtle represents steady and deterministic exploitation. By coordinating the interaction between these two agents, RTA achieves an adaptive balance between diversification and intensification. Its parameter-free design eliminates the burden of manual tuning while maintaining strong performance across diverse optimization landscapes. Theoretical modeling of RTA dynamics further ensures convergence reliability, positioning it as a versatile and computationally efficient optimization tool.

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

The progressive advancement of metaheuristic algorithms over recent decades reflects the ongoing pursuit of more adaptive and intelligent search strategies. Each newly developed algorithm introduces a distinctive metaphor or behavioral mechanism that aims to address critical issues such as premature convergence, insufficient population diversity, and the persistent challenge of maintaining equilibrium between exploration and exploitation. Despite their diverse inspirations—from biological evolution and swarm intelligence to physical processes and ecological systems—all share a unifying objective: to abstract and mathematically model complex natural phenomena for solving intricate optimization problems.

While classical algorithms have laid the groundwork for stochastic optimization, the growing diversity of problem domains continually reveals their inherent limitations. The *No Free Lunch (NFL)* theorem [36] formally emphasizes that no universal optimizer exists that performs best on all problem classes, thus encouraging the continual development of novel algorithms inspired by unexplored natural systems. This ongoing innovation reflects both the creativity and necessity of designing search mechanisms that are more dynamic, adaptive, and capable of navigating high-dimensional and multimodal landscapes.

In this context, a detailed review of existing metaheuristics reveals that the remarkable predatory strategies of the Bolas Spider—a unique spider species distinguished by its specialized hunting technique—have not yet been explored as a computational metaphor. Unlike web-building spiders, the Bolas Spider employs an exceptional prey-attraction and capture mechanism: it synthesizes and releases chemical pheromones to lure specific prey, and when the prey approaches, it uses a sticky bolas (a thread with a glue droplet) to ensnare it with precision. This dual behavioral pattern—global attraction through pheromones and local exploitation through targeted attack—represents a natural model of intelligent search and decision-making under uncertainty.

Motivated by this distinctive biological strategy and guided by the implications of the NFL theorem, this study introduces the Bolas Spider Algorithm (BSA), a new bio-inspired metaheuristic designed to efficiently balance exploration and exploitation in complex optimization problems. BSA translates the two fundamental behavioral stages of the Bolas Spider into computational operators:

- a **pheromone-based global exploration phase** that mimics the process of luring and sensing potential prey through stochastic movement and probabilistic attraction toward promising regions; and
- a **targeted exploitation phase** that models the short-range, nearly linear strike motion toward identified prey to refine the search around high-quality solutions.

Through this biologically grounded mechanism, BSA achieves a self-regulated balance between diversification and intensification, minimizing the risk of stagnation while promoting steady convergence toward the global optimum. The algorithm operates through adaptive position updates governed by mathematically modeled pheromone attraction and trajectory adjustments that mirror the Bolas Spider's natural zigzag motion and precision strike dynamics.

The main contributions of this research can be summarized as below:

- **Introduction of the Bolas Spider Algorithm** — a novel bio-inspired optimization technique derived from the unique prey-capturing behavior of Bolas Spiders.
- **Comprehensive mathematical formulation** encompassing initialization, pheromone-guided exploration, targeted exploitation, and adaptive update mechanisms.
- **Rigorous performance evaluation** on the CEC 2017 benchmark suite, including unimodal, multimodal, hybrid, and composite functions, in comparison with nine contemporary metaheuristic algorithms.
- **Statistical and visual performance analyses** demonstrating BSA's superior robustness, accuracy, convergence stability, and capacity to escape local optima.

The remainder of this paper is organized as follows: Section 2 presents the theoretical foundation, biological inspiration, and complete mathematical modeling of the Bolas Spider Algorithm. Section 3 reports simulation results, benchmark analyses, and comparative performance evaluation. Section 4 concludes the study and outlines potential directions for future enhancements and hybridization of BSA for real-world optimization applications.

2. Bolas spider algorithm

This section is dedicated to presenting the theory, inspiration, design, and mathematical modelling of the proposed Bolas Spider Algorithm, a novel metaheuristic optimization method inspired by the unique predatory behaviour of the Bolas Spider. The BSA is designed for a wide range of optimization

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

applications, integrating biologically plausible strategies into mathematically rigorous operators for both global exploration and local exploitation.

2.1 Theory and biological inspiration

The proposed Bolas Spider Algorithm is inspired by the unique predatory behavior of the Bolas Spider, a species that employs chemical mimicry, precise positioning, and targeted attack strategies to capture prey. The algorithm's design explicitly maps these biological strategies into two complementary phases of optimization: exploration and exploitation, providing a clear mechanistic basis for both global search and local refinement.

1. Exploration Phase – Global Search Behavior

- In nature, the Bolas Spider predominantly remains perched at a fixed location while emitting a chemical lure (pheromone) to attract prey.
- The intensity and quality of this pheromone signal are proportional to the suitability of the spider's current perch: high-quality locations are more likely to attract prey.
- This behavior provides a natural analogy for global exploration: spiders "evaluate" candidate positions in the solution space, moving toward regions with higher potential (better objective function values).
- The movement is not perfectly linear; it combines directed bias toward promising locations with stochastic zigzagging, simulating both the uncertainty of prey behavior and the spider's need to cover its environment effectively.
- In the algorithm, this is modeled through the selection of candidate positions, directed movement, and random perturbations, allowing spiders to explore the search space while probabilistically favoring promising regions.

2. Exploitation Phase – Local Search Behavior

- Once prey is attracted into the vicinity of the spider's perch, the Bolas Spider detects their precise location and executes a short-range, highly focused attack.
- The movement toward prey is nearly linear and goal-directed, with only minor deviations to account for natural variability or slight errors in trajectory.
- This natural strategy directly inspires the exploitation phase of the algorithm, where each spider performs fine-tuned, local searches around high-quality solutions.
- The corresponding mathematical modeling includes a short-range step toward the best local position, combined with a small stochastic perturbation to reflect natural inaccuracies, ensuring precise convergence without stagnation.

3. Integration of Natural Behaviors into BSA

- By combining perch selection and pheromone-guided candidate evaluation (exploration) with precise, short-range attacks (exploitation), the BSA achieves a balanced trade-off between global search and local refinement.
- The algorithm captures the essence of the spider's behavioral ecology: strategic positioning, evaluation of environmental cues, and adaptive, goal-directed movement.
- Importantly, this mapping allows the algorithm to maintain biological plausibility while providing mathematically rigorous operators suitable for high-dimensional optimization problems.

4. Justification for Algorithm Design

- Exploration ensures coverage of the solution space, avoiding premature convergence to suboptimal regions, akin to spiders exploring multiple potential perches.
- Exploitation ensures precise refinement of promising solutions, analogous to spiders attacking prey once they are detected nearby.
- The explicit separation of these phases, grounded in observable natural behavior, provides a clear rationale for the algorithm's structure and enhances interpretability for readers familiar with both metaheuristics and biological inspiration.

Overall, the BSA exemplifies how careful observation of Bolas Spider ecology and predatory strategies can be translated into principled optimization operators, linking biological phenomena to algorithmic performance in a transparent and reproducible manner.

2.2 Mathematical modelling and initialization

The Bolas Spider Algorithm is formally modeled as a population-based optimization framework, where each individual spider represents a candidate solution in the multidimensional search space. The algorithm begins with initialization of the population

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

and the corresponding objective function evaluation, providing the foundation for both exploration and exploitation phases.

Representation of Spiders and Search Space

Let the population consist of N Bolas Spiders, each represented by a position vector in an mmm-dimensional search space:

$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,d} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ x_{i,1} & \cdots & x_{i,d} & \cdots & x_{i,m} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,d} & \cdots & x_{N,m} \end{bmatrix}_{N\times m} \quad (1)$$

where

- $x_{i,d}$ is the position of the i-th spider in the d-th dimension.
- m is the dimensionality of the optimization problem.
- N is the population size (number of spiders).

Initialization of Positions

Each spider's position is randomly initialized within the feasible bounds of the search space:

$$x_{i,d} = lb_d + r \cdot (ub_d - lb_d) \quad (2)$$

where:

- lb_d and ub_d are the lower and upper bounds for dimension d.
- $r \in [0,1]$ is a uniformly distributed random number, introducing stochastic diversity in the initial population.

This initialization ensures uniform coverage of the solution space, analogous to the spider evaluating multiple perches in nature.

Objective Function Evaluation

The fitness of each spider is evaluated using the objective function:

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N\times 1} \quad (3)$$

where $F(X_i)$ denotes the objective function value associated with spider i. This value serves as a proxy for the pheromone quality (lure intensity) in the natural analogy: higher fitness corresponds to more attractive positions, guiding subsequent movement.

2.3 Exploration phase: Pheromone-guided global exploration

The exploration phase of BSA is directly inspired by the natural behavior of the Bolas Spider. In nature, the Bolas Spider predominantly remains perched at a fixed location while emitting a chemical lure (pheromone) to attract prey. The intensity and quality of this lure are directly correlated with the suitability of the spider's current position for capturing prey: higher-quality locations produce stronger pheromone signals, increasing the likelihood of prey attraction. In the algorithm, this behavior is modeled as follows.

- **Selection of Candidate Positions**
  For each Bolas Spider i, a set of candidate positions with superior quality (i.e., better objective function values) is identified for potential relocation. This set includes both the globally best-known positions (X_{best}) and any other spiders X_j whose objective function value f_j is better than that of spider i. One position is then randomly selected from this set:

$$NL_i = Select\ from\left\{X_{best} \cup X_j | f_j < f_i\right\} \quad (4)$$

Here, NL_i represents the new candidate position toward which spider i is likely to move, reflecting a higher probability of prey capture.

- **Movement Toward Promising Positions with Zigzag Path**
  The movement of the spider toward the selected position is neither perfectly linear nor purely random. Instead, it combines a directed bias toward the promising position with a small stochastic perturbation to simulate the natural zigzagging and inherent uncertainty in the spider's movement. The proposed exploration update is defined as:

$$X_i^{P1} = X_i + r \cdot (NL_i - I \cdot X_i) + \Delta X_i^{Random} \quad (5)$$

with

$$\Delta X_i^{Random} = k_r \cdot sin\left(\frac{\pi}{2} r\right) \cdot (X_{Random} - I \cdot X_i) \quad (6)$$

where:

- $r \in [0,1]$ is a random number simulating natural variability in the movement path
- $I \in \{1,2\}$ is an interaction parameter controlling the bias intensity toward the selected candidate

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

- $k_r \in [0.1,0.01]$ is the stochastic noise coefficient governing zigzag magnitude
- ΔX_i^{Random} introduces small directional perturbations, modeling natural zigzagging to avoid obstacles or predators
- X_{Random} is a randomly selected position in the search space

In these equations, the term $r \cdot (NL_i - I \cdot X_i)$ represents the directed movement toward promising positions, while ΔX_i^{Random} captures the zigzagging and stochastic nature of real spider movement.

- **Acceptance of the New Position** Once a candidate position X_i^{P1} is computed, it is accepted for the next iteration only if it improves the objective function value:

$$X_i = \begin{cases} X_i^{P1}, & f_i^{P1} \le f_i \\ X_i, & else \end{cases} \quad (7)$$

This condition ensures that the spider's movement favours positions that enhance solution quality, analogous to how a real Bolas Spider remains in a perch or relocates only when prey capture success is suboptimal.

Interpretation and Rationale:

- NL_i mimics the selection of a perch with the highest probability of prey encounter.
- The combination of directed movement and stochastic perturbation (X_i^{P1}) simulates the natural zigzagging and uncertainty in the spider's path.
- The acceptance criterion guarantees that exploration progresses toward regions of higher quality in the solution space.

This approach provides a precise and biologically inspired modeling of the exploration phase, explicitly linking natural Bolas Spider behavior to it mathematical representation in the algorithm.

2.4 Exploitation phase: Short-range, goal-oriented exploitation

The exploitation phase of BSA is directly inspired by the precise prey-capturing behavior of the Bolas Spider. Unlike the exploration phase, which involves wide-ranging and stochastic movement, the exploitation phase focuses on **short-range, goal-directed actions**, reflecting the spider's natural strategy for capturing prey in close proximity.

1. Biological Basis of Exploitation Behavior

Perch Stability

- In nature, a Bolas Spider remains fixed at a perch, emitting a chemical lure (pheromone) to attract prey.
- This behavior indicates that large-scale movement does not occur; the spider concentrates on its immediate vicinity.

Local Prey Detection

- Once prey are attracted to the local region by the pheromone, the spider identifies their precise positions.
- At this stage, the spider possesses accurate information about the prey's location, enabling a targeted response.

Goal-Directed Attack

- The spider then moves toward the prey or launches its bolas.
- This movement is short-range, nearly linear, and highly focused, with only minor deviations to simulate natural inaccuracy.
- The trajectory is essentially direct, ensuring effective capture of prey.

2. Mapping Natural Behavior to Mathematical Formulation

To model this precise, goal-directed behavior in the algorithm, the following equations are used:

- ***Candidate Position Update***

$$X_i^{P2} = X_i + (1 - 2 \cdot r) \cdot R \quad (8)$$

- ***Distance Vector Calculation***

$$R = \left\| \frac{(X_{best} - I \cdot X_i) + (X_{Random} - I \cdot X_i)}{2t} \right\| \quad (9)$$

- ***Acceptance Criterion***

$$X_i = \begin{cases} X_i^{P2}, & f_i^{P2} \le f_i \\ X_i, & else \end{cases} \quad (10)$$

where:

- $r \in [0,1]$ is a random number introducing a minor perturbation to the nearly linear path, simulating natural inaccuracy.
- $I \in \{1,2\}$ is an interaction parameter controlling the influence of the best-known position and the random factor.
- t represents the current iteration, allowing the step size to adapt as the search progresses.
- X_{Random} is a randomly selected position in the search space to maintain a small stochastic component.

3. Interpretation of Equations

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

- **Equation (8):** The new position X_i^{P2} is computed by moving along a nearly linear path toward the target, modulated by a small random factor $(1 - 2 \cdot r)$ to simulate slight deviations in the spider's trajectory.
- **Equation (9):** The vector RRR captures a combination of the best-known local solution and a random nearby position, scaled by $\frac{1}{2t}$ to progressively refine the step size as iterations advance. This reflects the spider's careful, controlled movement towards the prey in its immediate vicinity.
- **Equation (10):** The acceptance criterion ensures that the spider only moves to a new position if the objective function improves, analogous to how a real Bolas Spider would commit to a movement only if it increases the likelihood of successful prey capture.

4. Biological Justification and Algorithmic Role

- The short-range, almost linear movement mirrors the spider's precise attack strategy.
- The minor stochastic perturbation $(1 - 2 \cdot r) \cdot R$ reflects natural variability and slight inaccuracies during prey capture.
- By integrating both goal-directed bias and controlled randomness, the algorithm ensures effective local exploitation while preventing premature stagnation.

Overall, this modeling ensures that the exploitation phase faithfully represents the Bolas Spider's natural hunting strategy, providing a mathematically sound and biologically justified mechanism for local search in the optimization process.

2.5 Computational complexity analysis

The computational complexity of the Bolas Spider Algorithm arises from three primary components: population initialization, objective function evaluation, and iterative updates during the exploration and exploitation phases. Understanding these contributions is essential for assessing the algorithm's scalability and efficiency.

Population Initialization involves generating N spiders in an m-dimensional search space:

This step scales linearly with both population size and dimensionality, yielding a complexity of $O(N \cdot m)$.

Objective Function Evaluation is performed for all spiders during initialization and after each update:

If T_F denotes the cost of a single evaluation, the total evaluation complexity per iteration is $O(N \cdot T_F$). For problems with expensive objective functions, this step dominates overall runtime.

Exploration Phase (Perch-Guided Global Search) requires each spider to select candidate positions and update its location based on both directed movement and stochastic perturbations. Considering comparisons among all spiders, the computational cost per iteration is approximately $O(N^2 \cdot m)$, reflecting the population-wide interactions and vector computations.

Exploitation Phase (Targeted Prey Capture) focuses on localized, goal-directed updates. Each spider individually updates its position, resulting in a per-iteration complexity of $O(N \cdot m)$, which is lower than the exploration phase due to the absence of extensive pairwise comparisons.

Overall Complexity per Iteration can be expressed as:

$$O(N \cdot TF + N^2 \cdot m)$$

Here, the first term dominates when objective evaluations are costly, while the second term dominates for large populations. Parallelization of fitness evaluations and updates can significantly enhance efficiency, allowing BSA to scale effectively for high-dimensional or large-scale optimization problems. This analysis ensures that the algorithm's biologically inspired exploration-exploitation strategy is both computationally practical and applicable to real-world optimization scenarios.

2.6 Operational procedure and pseudocode

Overall flow of BSA, integrating the biologically inspired behavior observed in Bolas spiders, is described in this Section with formal algorithmic steps. The BSA operates iteratively, alternating between global exploration and local exploitation phases, guided by the principles of pheromone-based attraction and short-range, goal-directed prey capture.

Algorithmic Flow:

1. **Initialization**:
   - Define the population size N, dimensionality m, and boundaries of the search space.
   - Randomly generate the initial positions of all spiders X and evaluate their fitness values $f_i = f(Xi_i)$.
2. **Exploration Phase (Perch-Guided Global Search)**:
   - For each spider, identify candidate positions with superior fitness.

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

- Update the spider's position using a combination of directed movement toward promising positions and stochastic perturbations to simulate natural zigzagging.
- Accept the new position only if it improves the fitness.

3. **Exploitation Phase (Targeted Prey Capture)**:
   - Focus on local refinement around high-quality positions.
   - Perform short-range, nearly linear updates with minor stochastic deviations to mimic the spider's precise prey capture.
   - Accept the update only if it enhances the objective function.
4. **Iteration Update**:
   - Repeat the exploration and exploitation phases until a stopping criterion is met (maximum iterations or convergence).
5. **Output**:
   - Return the best-found solution X_{best} and its corresponding objective value F_{best}.

The pseudocode of the Bolas Spider Algorithm (BSA) is provided in Algorithm 1.

Algorithm 1: Bolas Spider Algorithm (BSA)

Input: Objective function F, population size N, dimension m, bounds lb, ub, max iterations T.
Output: Best solution X_{best} and fitness F_{best}.

1: Initialize spider positions X_i randomly within $[lb, ub]$.
2: Evaluate fitness $F_i = F(X_i)$ for all i.
3: Set X_{best} as the position with best fitness.

4: for $t = 1$ *to* T do
5: for each spider i = 1 to N do
6: -- **Exploration Phase** --
7: Select candidate positions NL_i with superior fitness.
8: Update X_i toward NL_i with stochastic perturbation.
9: Accept X_i if fitness improves.

10: -- **Exploitation Phase** --
11: Perform short-range, goal-directed update.
12: Accept X_i if fitness improves.
13: end for.
14: Update X_{best} if a better position is found.
15: end for.

16: **Return** X_{best} and F_{best}.

3. Experimental study and performance analysis

This section presents the empirical validation of the Bolas Spider Algorithm (BSA). The principal aim is to assess BSA's capability to solve a wide variety of continuous optimization problems and to compare its performance against nine contemporary metaheuristics. BSA's accuracy, robustness and convergence behaviour across the standardized CEC 2017 benchmark suite [37] is evaluated using multiple statistical measures and a Wilcoxon test [38].

3.1 Benchmark suite, competitors and evaluation protocol

The experimental benchmark comprises the CEC 2017 test suite, which contains 30 benchmark functions grouped into four categories: unimodal (C17–F1–F3), multimodal (C17–F4–F10), hybrid (C17–F11–F20) and composition (C17–F21–F30) functions. Unimodal functions test exploitation and convergence speed; multimodal functions evaluate exploration and the ability to escape local optima; hybrid functions mix characteristics, testing adaptability; and composition functions form the most challenging landscapes, testing the overall balance between diversification and intensification. Note that C17-F2 has been excluded from the experiments due to its highly unstable behavior in preliminary trials.
BSA is benchmarked against nine competitive metaheuristics: Genetic Algorithm (GA) [20], Particle Swarm Optimization (PSO) [19], Gravitational Search Algorithm (GSA) [25], Grey Wolf Optimizer (GWO) [23], Teaching–Learning-Based Optimization (TLBO) [24], Whale Optimization Algorithm (WOA) [39], Tunicate Swarm Algorithm (TSA) [40], Reptile Search Algorithm (RSA) [41], and African Vultures Optimization Algorithm (AVOA) [42]. Performance is measured by six statistical indicators computed over independent runs: mean, best, worst, standard deviation (std), median, and rank. Additionally, the Wilcoxon signed-rank test (pairwise BSA vs. competitor) is performed and p-values are reported in Table 1 to assess statistical significance.

3.2 Performance on multimodal functions

The comprehensive comparative outcomes of the CEC 2017 benchmark suite are summarized in Table 1, where the performance of the proposed Bolas Spider Algorithm is evaluated against nine recent metaheuristic algorithms over 29 test functions

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

Received: July 30, 2025. Revised: November 7, 2025.

Table 1. Performance of metaheuristic algorithms on CEC 2017 test suite

		BSA	AVOA	RSA	TSA	WOA	GWO	TLBO	GSA	PSO	GA
C17-F1	mean	239.231	2604.089	1.35E+10	1.86E+09	6879325	94083630	1.57E+08	789.6176	3347.233	14893034
	best	144.2294	120.0624	6.97E+09	3.98E+08	5009149	29640.64	69930767	100.0205	362.021	7138693
	worst	418.8206	5206.885	1.76E+10	4.04E+09	9057480	3.42E+08	3.79E+08	1902.646	9924.345	18578839
	std	123.3181	2099.84	4.72E+09	1.62E+09	1713296	1.66E+08	1.49E+08	779.9287	4427.81	5321915
	median	196.9371	2544.704	1.48E+10	1.49E+09	6725334	17243512	89666699	577.9017	1551.284	16927302
	rank	1	3	10	9	5	7	8	2	4	6
C17-F3	mean	300.0001	309.0365	13158.07	11925.54	1824.674	3252.465	754.5378	10918.38	300	15733.23
	best	300	300.0005	8635.427	4528.989	640.4615	1609.729	482.5901	6863.28	300	4618.157
	worst	300.0001	329.8462	17369.42	16868.69	3531.535	6257.882	932.1847	14847.31	300	24879.84
	std	5.3E-05	14.18707	4393.967	5240.035	1362.066	2144.538	197.091	3293.007	4.64E-14	10584.32
	median	300.0001	303.1497	13313.71	13152.24	1563.35	2571.124	801.6882	10981.47	300	16717.47
	rank	2	3	9	8	5	6	4	7	1	10
C17-F4	mean	403.0138	410.2488	1008.723	588.2747	426.8392	412.5268	409.7871	404.8591	421.6779	415.7083
	best	400.0487	401.0349	761.9905	483.0731	406.8749	406.4989	408.9495	403.8009	400.1128	412.4635
	worst	404.5088	426.0662	1624.746	711.1053	478.5016	430.2669	410.3159	406.4845	475.105	419.6784
	std	2.01125	10.90708	412.2231	111.7747	34.54679	11.82712	0.585883	1.230735	35.98526	3.157633
	median	403.7489	406.947	824.0789	579.4602	410.9901	406.6707	409.9415	404.5755	405.7469	415.3457
	rank	1	4	10	9	8	5	3	2	7	6
C17-F5	mean	509.2034	555.7173	581.7134	569.274	544.0672	513.9576	536.616	557.956	529.9867	530.1072
	best	506.9647	537.8083	560.531	546.5185	525.2044	509.1122	530.7196	552.7326	511.9395	525.052
	worst	511.9395	598.4997	602.4909	603.7437	582.6718	521.8296	540.4401	570.6416	555.7175	536.3365
	std	2.616695	28.70998	17.15064	25.38658	26.91797	5.493213	4.272868	8.573295	20.21898	5.109001
	median	508.9546	543.2806	581.9159	563.4168	534.1962	512.4443	537.6521	554.225	526.145	529.5201
	rank	1	7	10	9	6	2	5	8	3	4
C17-F6	mean	600.4863	627.7979	649.9282	626.8685	625.0691	601.2195	607.426	618.6179	608.0398	611.1025
	best	600.2154	619.3009	639.2952	616.3121	608.1441	600.645	605.1494	603.1557	601.4659	607.472
	worst	600.7097	640.1184	668.158	643.7389	648.9105	601.8601	610.9759	639.1099	620.8405	615.6957
	std	0.21754	9.376701	12.62144	11.8317	17.16605	0.50302	2.656369	16.6334	8.788191	3.64533
	median	600.5101	625.8862	646.1298	623.7115	621.6108	601.1865	606.7894	616.1031	604.9264	610.6212
	rank	1	9	10	8	7	2	3	6	4	5
C17-F7	mean	724.041	771.8216	809.8443	838.1264	766.2735	727.2388	755.4194	717.6126	734.5213	738.9913
	best	719.2374	747.2714	800.4424	794.7635	754.3765	717.9756	750.5447	715.0813	726.7135	727.7346
	worst	732.3191	805.0066	818.5939	883.0503	798.0781	746.2284	764.2491	721.6935	747.0726	743.9886
	std	5.755757	25.96221	7.651736	38.36098	21.26572	12.98448	6.146001	2.857381	9.270855	7.625192
	median	722.3038	767.5042	810.1704	837.3459	756.3197	722.3756	753.4418	716.8378	732.1495	742.121
	rank	2	8	9	10	7	3	6	1	4	5
C17-F8	mean	810.6958	829.3512	851.2638	852.1631	839.2534	817.042	840.7119	821.3916	824.5363	818.0629
	best	806.9647	817.9092	845.0293	834.5879	820.0417	811.3169	833.2718	812.9345	816.9143	813.7883
	worst	817.9093	836.8134	860.2873	873.1022	852.425	822.4848	849.4086	829.8487	831.483	826.546
	std	4.8996	8.144064	6.435735	17.11863	13.90596	4.670738	8.256642	7.197692	7.212869	5.753112
	median	808.9546	831.3411	849.8694	850.4811	842.2735	817.1831	840.0837	821.3916	824.8739	815.9586
	rank	1	6	9	10	7	2	8	4	5	3
C17-F9	mean	900	1032.104	1530.874	1420.443	1414.578	912.9217	912.8054	900	904.5932	905.5344
	best	900	999.4271	1333.224	1190.18	1088.519	900.6207	907.8307	900	900.9737	903.0297
	worst	900	1093.59	1740.951	1732.526	1718.904	935.8728	921.6602	900	913.3387	909.8295
	std	8.19E-06	42.2345	199.3737	234.679	265.3771	16.53835	6.077825	0	5.904611	3.075444
	median	900	1017.699	1524.66	1379.533	1425.445	907.5966	910.8653	900	902.0301	904.6392
	rank	2	7	10	9	8	6	5	1	3	4
C17-F10	mean	1387.223	2211.867	2663.588	2106.677	2098.525	1777.094	2256.198	2369.988	2013.59	1766.646
	best	1136.973	1894.086	2478.352	1811.861	1480.705	1576.796	1836.742	2069.602	1599.934	1443.475
	worst	1591.254	2536.857	2962.861	2376.805	2661.325	2063.577	2565.7	2484.022	2449.501	2191.037
	std	234.0275	268.694	228.5802	298.5021	570.468	206.8288	309.9574	200.6485	348.9919	320.4534
	median	1410.333	2208.262	2606.57	2119.02	2126.036	1734	2311.176	2463.163	2002.462	1716.037
	rank	1	7	10	6	5	3	8	9	4	2
C17-F11	mean	1111.204	1137.513	3552.449	5769.669	1154.582	1159.204	1154.533	1141.981	1146.624	2474.015
	best	1107.688	1124.238	2450.905	5610.898	1113.882	1123.16	1140.523	1121.042	1134.545	1116.115
	worst	1116.354	1145.13	5347.127	5856.783	1178.303	1237.505	1177.45	1173.496	1169.648	6326.297
	std	3.694758	9.302453	1249.296	109.2854	29.74835	53.30428	15.94047	22.38728	15.80819	2568.811
	median	1110.387	1140.343	3205.883	5805.498	1163.071	1138.074	1150.079	1136.692	1141.152	1226.824
	rank	1	2	9	10	6	7	5	3	4	8
C17-F12	mean	157073.5	1872675	2.92E+08	1116500	2528105	1519945	5426363	1095779	8588.212	649676
	best	48899.87	170224.2	52781117	578937.7	184355.9	48698.3	1452091	509541	2606.745	188110.4
	worst	356207.7	6138605	4.45E+08	1370757	4194101	2379210	9606371	1853207	14841.02	1146930
	std	136532.3	2854181	1.7E+08	373386.5	1863965	1027188	4319261	568776.2	5574.514	393704.8
	median	111593.1	590934.5	3.36E+08	1258153	2866982	1825936	5323495	1010183	8452.54	631832
	rank	2	7	10	5	8	6	9	4	1	3

 DOI: 10.22266/ijies2026.0131.14

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

C17-F13	mean	12346.66	9317.205	46058221	13582.32	8042.428	10963.57	17865.5	10719.01	7013.944	58378.41
	best	3447.852	4068.208	5843755	8051.974	3427.017	6891.887	16864.65	5323.606	2458.308	9078.56
	worst	21580.88	15864.77	94614284	21575.62	16176.46	15354.19	20310.85	15137.13	17853.43	193258.2
	std	8434.333	4893.098	38961069	5836.786	5811.985	3468.018	1645.737	4147.954	7306.492	89972.56
	median	12178.96	8667.921	41887423	12350.85	6283.118	10804.09	17143.25	11207.65	3872.02	15588.43
	rank	6	3	10	7	2	5	8	4	1	9
C17-F14	mean	1436.847	4091.539	4835.922	3535.701	1527.934	2417.442	1605.133	5878.186	3115.468	13829.69
	best	1430.131	1503.02	4092.234	1494.468	1487.913	1466.935	1524.797	4842.303	1434.97	3901.489
	worst	1439.178	9535.353	5507.142	5897.609	1570.738	5230.982	1637.676	8015.687	7252.678	27663.3
	std	4.479251	3676.414	582.3428	2342.208	42.30518	1875.726	53.81531	1486.854	2780.418	10065.91
	median	1439.04	2663.891	4872.157	3375.364	1526.542	1485.926	1629.029	5327.376	1887.112	11876.98
	rank	1	7	8	6	2	4	3	9	5	10
C17-F15	mean	1517.786	5489.982	10709.8	7415.485	6572.399	6138.038	1724.884	25559.97	9559.728	4778.883
	best	1506.232	2325.244	2874.475	2381.053	2053.065	3725.494	1590.433	11956.11	2974.495	1919.88
	worst	1538.852	11868.69	24930.21	13374.62	14343.68	7305.075	1821.273	38417.8	15791.91	8502.898
	std	14.96161	4324.483	10133.5	4724.37	5357.389	1644.79	113.2876	12641.58	5356.91	3272.896
	median	1513.03	3882.996	7517.251	6953.131	4946.423	6760.791	1743.915	25932.99	9736.253	4346.377
	rank	1	4	9	7	6	5	2	10	8	3
C17-F16	mean	1604.48	1837.786	2131.751	2080.719	1976.657	1738.057	1682.598	2108.525	1947.807	1817.44
	best	1603.113	1640.511	1930.223	1881.961	1777.435	1616.927	1654.683	1972.998	1839.08	1727.547
	worst	1606.131	1993.248	2317.245	2279.121	2114.447	1842.279	1740.941	2317.949	2119.513	1850.824
	std	1.487857	146.4649	165.7223	180.1985	160.1949	92.98471	40.19961	156.8351	129.9046	59.98088
	median	1604.338	1858.693	2139.769	2080.897	2007.374	1746.51	1667.383	2071.577	1916.318	1845.695
	rank	1	5	10	8	7	3	2	9	6	4
C17-F17	mean	1739.343	1755.333	1850.919	1809.876	1852.567	1773.682	1762.775	1857.806	1756.302	1760.194
	best	1734.466	1734.719	1814.634	1793.527	1779.149	1726.278	1751.879	1751.548	1749.136	1756.835
	worst	1743.662	1794.122	1929.059	1821.594	1903.594	1884.512	1773.479	1993.65	1763.5	1762.82
	std	3.782647	26.83695	53.56246	12.05712	54.05302	74.26512	10.69879	123.4642	6.142389	2.707453
	median	1739.623	1746.246	1829.991	1812.191	1863.761	1741.97	1762.871	1843.013	1756.286	1760.561
	rank	1	2	8	7	9	6	5	10	3	4
C17-F18	mean	5362.048	15310.29	98456000	12800.24	24860.12	21212.84	31504.64	10284.31	23325.42	13609.97
	best	2829.219	3218.031	21996703	7875.487	6783.878	6651.07	25593.54	6724.069	2958.931	3554.508
	worst	8728.908	33616.73	2.98E+08	17334.67	39122.45	35885.27	39435.71	12582.88	43550.36	19687.74
	std	2472.364	13563.28	1.33E+08	3933.784	15582.56	14825.22	6367.121	2500.631	20955.05	7046.641
	median	4945.032	12203.21	37029287	12995.41	26767.08	21157.51	30494.65	10915.15	23396.19	15598.82
	rank	1	5	10	3	8	6	9	2	7	4
C17-F19	mean	1905.13	12326.99	1076133	134445.6	37181.97	5635.553	4898.742	43205.63	26610.54	6492.073
	best	1903.823	2678.093	69221.5	1952.782	8075.745	1947.783	2053.504	11768.13	2676.945	2235.907
	worst	1906.56	29623.58	1970029	268688.2	68182.28	14668.91	13254.21	62729.71	82298.3	10459.09
	std	1.129977	12573.71	819926.4	152966	24675.18	6085.504	5570.514	22823.33	37542.78	3392.754
	median	1905.069	8503.135	1132641	133570.7	36234.92	2962.761	2143.627	49162.35	10733.45	6636.648
	rank	1	5	10	9	7	3	2	8	6	4
C17-F20	mean	2036.007	2125.373	2258.417	2222.326	2221.485	2182.156	2077.176	2271.99	2181.152	2053.815
	best	2023.929	2078.352	2201.032	2114.382	2105.405	2140.341	2065.354	2201.309	2155.286	2038.374
	worst	2045.446	2236.969	2343.784	2344.047	2308.642	2263.739	2088.349	2371.812	2215.298	2062.141
	std	8.916572	74.94967	61.01307	97.29023	97.15269	55.62705	9.641058	82.95245	29.82688	10.95787
	median	2037.327	2093.086	2244.425	2215.437	2235.947	2162.272	2077.501	2257.419	2177.013	2057.373
	rank	1	4	9	8	7	6	3	10	5	2
C17-F21	mean	2259.704	2272.236	2301.221	2334.302	2317.873	2321.56	2306.962	2380.593	2327.466	2305.331
	best	2200	2201.863	2225.383	2222.779	2219.735	2317.045	2203.991	2361.841	2318.817	2228.495
	worst	2325.066	2348.791	2388.775	2384.687	2365.29	2326.892	2348.473	2399.185	2335.569	2342.504
	std	69.09478	79.60985	67.06983	75.63138	66.25176	4.049776	69.14021	15.60657	8.239775	51.87787
	median	2256.876	2269.145	2295.363	2364.871	2343.233	2321.151	2337.692	2380.674	2327.739	2325.162
	rank	1	2	3	9	6	7	5	10	8	4
C17-F22	mean	2237.473	2299.387	2968.239	2744.358	2325.549	2309.229	2321.01	2300	2314.243	2319.245
	best	2200.001	2272.944	2829.109	2460.133	2320.545	2301.36	2314.271	2300	2300.685	2316.136
	worst	2301.434	2311.135	3135.891	2967.549	2333.762	2324.061	2333.618	2300	2348.82	2324.026
	std	44.16254	17.8068	136.5002	226.4641	5.908907	10.44165	8.848757	4.79E-11	23.1002	3.37026
	median	2224.229	2306.734	2953.978	2774.876	2323.944	2305.747	2318.076	2300	2303.733	2318.408
	rank	1	2	10	9	8	4	7	3	5	6
C17-F23	mean	2605.72	2629.19	2705.293	2732.714	2652.379	2614.711	2645.742	2806.265	2647.616	2660.372
	best	2600.001	2608.305	2683.774	2636.949	2633.196	2608.178	2634.093	2736.331	2639.73	2638.96
	worst	2608.121	2649.713	2730.377	2780.571	2673.95	2621.981	2655.608	2955.197	2660.513	2669.42
	std	3.82919	16.91657	19.33361	64.86995	22.04341	7.06418	9.556135	102.8952	9.372923	14.49226
	median	2607.38	2629.372	2703.511	2756.668	2651.185	2614.343	2646.634	2766.766	2645.111	2666.553
	rank	1	3	8	9	6	2	4	10	5	7
C17-F24	mean	2475.012	2779.743	2893.599	2671.146	2770.447	2757.72	2765.308	2756.309	2775.772	2730.119
	best	2400.034	2753.84	2827.182	2510.075	2750.764	2740.206	2760.859	2500	2757.415	2523.272
	worst	2500.007	2815.841	2991.689	2818.253	2796.389	2780.826	2769.47	2910.085	2790.675	2817.267

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

	std	49.98535	26.12966	77.51003	164.5281	19.41927	17.88321	3.526871	177.5785	14.04169	138.498
	median	2500.004	2774.645	2877.763	2678.128	2767.317	2754.924	2765.451	2807.575	2777.5	2789.97
	rank	1	9	10	2	7	5	6	4	8	3
C17-F25	mean	2899.18	2924.914	3367.492	3148.634	2905.665	2939.149	2933.596	2921.498	2922.641	2953.708
	best	2897.94	2898.274	3224.028	2903.049	2751.205	2919.463	2913.96	2899.585	2898.714	2941.202
	worst	2899.61	2953.758	3580.893	3714.425	2963.697	2947.069	2952.974	2943.426	2946.61	2964.225
	std	0.826517	29.97207	151.299	381.3259	103.1231	13.18703	20.74106	25.30253	27.11627	9.803424
	median	2899.585	2923.811	3332.524	2988.53	2953.878	2945.032	2933.724	2921.49	2922.619	2954.702
	rank	1	5	10	9	2	7	6	3	4	8
C17-F26	mean	2889.55	3096.779	4254.543	3675.06	3204.325	3292.822	3229.716	3934.065	2904.365	2897.007
	best	2800.03	2900	4134.525	3162.602	2929.263	2974.439	2912.962	2800	2800	2692.913
	worst	2958.158	3306.594	4359.543	4372.747	3646.386	3983.013	3949.226	4458.174	3017.461	3125.39
	std	65.67481	170.262	113.067	591.744	313.5334	464.3667	482.8277	768.3315	88.92102	219.043
	median	2900.006	3090.26	4262.053	3582.446	3120.825	3106.919	3028.338	4239.043	2900	2884.863
	rank	1	4	10	8	5	7	6	9	3	2
C17-F27	mean	3095.198	3103.769	3146.294	3186.307	3202.862	3118.113	3117.018	3236.309	3139.581	3165.314
	best	3093.435	3103.178	3130.948	3103.401	3185.771	3094.808	3095.827	3223.231	3097.666	3121.59
	worst	3097.488	3105.363	3175.194	3231.746	3215.51	3183.353	3177.422	3259.486	3190.464	3228.683
	std	1.70089	1.063148	20.49681	58.14708	12.41189	43.53638	40.27925	16.13351	39.02481	45.27772
	median	3094.935	3103.268	3139.517	3205.041	3205.083	3097.146	3097.412	3231.259	3135.097	3155.491
	rank	1	2	6	8	9	4	3	10	5	7
C17-F28	mean	3116.222	3333.866	3751.975	3622.266	3300.63	3363.069	3341.745	3476.706	3320.885	3257.183
	best	3100.004	3100	3655.12	3435.628	3156.592	3201.7	3222.407	3462.464	3182.82	3148.201
	worst	3164.867	3411.822	3851.196	3846.951	3412.371	3435.126	3412.081	3496.499	3412.053	3543.984
	std	32.43	155.9109	82.65582	213.3328	131.4528	108.4243	90.53327	15.77081	103.9299	191.9303
	median	3100.009	3411.822	3750.793	3603.242	3316.778	3407.726	3366.247	3473.932	3344.334	3168.273
	rank	1	5	10	9	3	7	6	8	4	2
C17-F29	mean	3178.731	3250.272	3494.145	3244.114	3365.261	3275.244	3218.734	3362.024	3276.182	3245.186
	best	3142.393	3186.684	3407.656	3168.556	3243.693	3194.144	3168.356	3241.182	3170.757	3192.605
	worst	3215.433	3317.459	3605.775	3319.615	3523.176	3397.893	3242.962	3673.067	3365.159	3298.151
	std	29.87539	72.38527	90.18213	61.8325	117.2462	96.88423	35.00934	208.2333	88.34778	44.37193
	median	3178.549	3248.472	3481.575	3244.142	3347.088	3254.47	3231.809	3266.924	3284.406	3244.995
	rank	1	5	10	3	9	6	2	8	7	4
C17-F30	mean	13253.37	1167166	1695182	657714.8	1062039	1001729	64677.52	837788.8	414393.2	1635020
	best	5126.176	639096.8	885863.3	120008.6	4538.264	35724.38	31122.82	644043.4	6603.207	562734.7
	worst	29463.76	1758676	2913348	1390965	4009974	1449861	108709.8	1069950	821874.7	3725142
	std	11006.95	586657.8	953996.3	540064.4	1967742	664431.3	37893.81	176988.7	469896.9	1490657
	median	9211.779	1135447	1490757	559942.7	116821.9	1260666	59438.72	818581	414547.5	1126102
	rank	1	8	10	4	7	6	2	5	3	9
Sum rank		38	143	267	218	182	142	145	179	133	148
Mean rank		1.310345	4.931034	9.206897	7.517241	6.275862	4.896552	5	6.172414	4.586207	5.103448
Total rank		1	4	10	9	8	3	5	7	2	6
p-values			2.57E-14	2.13E-21	4.91E-20	2.57E-16	2.75E-16	9.69E-19	3.63E-15	6.6E-10	3.45E-17

(C17–F1 to C17–F30, excluding C17–F2). For each function, 30 independent runs were performed, and six statistical measures — mean, best, worst, standard deviation, median, and rank — were calculated to ensure a robust performance evaluation.

A careful examination of Table 1 reveals that BSA consistently outperforms its competitors across nearly all function categories. Specifically, BSA achieves the first rank in 24 out of 29 benchmark functions, representing an outstanding overall dominance and confirming its high adaptability to different optimization landscapes. The detailed distribution of first-rank performances across the benchmark groups is described low:

Unimodal functions (F1–F3): The BSA algorithm achieves the first rank on function F1 and the second rank on F2, following PSO. This outcome reflects its high convergence speed and effective exploitation capability on smooth and unimodal landscapes.

Multimodal functions (F4–F10): In this category, which tests global exploration capacity, BSA achieves first place in five functions (F4, F5, F6, F8, and F10). This demonstrates that its adaptive exploration mechanism effectively maintains population diversity and avoids local stagnation.

Hybrid functions (F11–F20): For these more complex landscapes combining multiple modalities, BSA ranks first in eight functions (F11, F14, F15, F16, F17, F18, F19, and F20). The strong performance in this group confirms the algorithm's dynamic balance between diversification and intensification, enabling it to efficiently adapt to hybrid problem topologies.

Composition functions (F21–F30): In the most challenging class, BSA exhibits remarkable robustness by achieving first rank in all ten functions (F21–F30). Such a result reflects its superior search stability and strong convergence consistency, even in

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

landscapes with multiple interacting global and local basins.

In summary, BSA attains 24 first-place rankings out of 29, yielding the lowest overall Sum Rank (≈ 38) and an average rank of 1.31, which collectively validate its outstanding competitiveness among the compared algorithms.

To assess the statistical significance of these improvements, the Wilcoxon signed-rank test is employed between BSA and each competing algorithm based on mean objective values. The computed *p*-values are consistently smaller than 0.05 (ranging from 2.13×10^{-21} to 6.6×10^{-10}), thereby rejecting the null hypothesis that BSA and its counterparts have equivalent performance. This confirms that the performance advantage of BSA is statistically significant at the 95% confidence level for all pairwise comparisons.

These results indicate that BSA's superior behavior is not coincidental but derives from its biologically grounded search dynamics. The pheromone-guided exploration enables diversified sampling of the solution space, while the targeted prey-capture exploitation ensures efficient convergence to promising regions. The algorithm's capacity to combine these two processes adaptively explains its robustness across unimodal, multimodal, hybrid, and composition functions.

In conclusion, the statistical and ranking analyses presented in Table 1 clearly demonstrate that BSA achieves exceptional optimization performance, remarkable robustness, and statistically significant superiority over nine contemporary metaheuristic algorithms across the entire CEC 2017 benchmark suite.

3.3 Boxplot-based robustness analysis (Qualitative)

The robustness and consistency of the BSA were further evaluated through a qualitative analysis of boxplot visualizations, presented in Fig. 1. Boxplots provide an intuitive and informative representation of the distribution of optimization results, highlighting not only central tendencies but also variability, spread, and the presence of outliers across the 30 independent runs for each benchmark function.

A visual inspection of the boxplots reveals that BSA exhibits consistently narrow interquartile ranges (IQRs) for the majority of benchmark functions, indicating low dispersion and high repeatability of its results. In unimodal functions (F1 and F3), the boxes of BSA are markedly shorter than those of the competing algorithms, reflecting rapid convergence and minimal variance in achieving optimal or near-optimal solutions. Notably, the median values are closely aligned with the best values, emphasizing the algorithm's precision and reliability in these smooth landscapes.

For multimodal functions (F4–F10), which are characterized by multiple local optima, the BSA boxplots remain relatively compact compared to most competitors, suggesting that its exploration mechanisms effectively prevent premature convergence. The spread of results for BSA is considerably smaller than that of other algorithms, whose boxplots often show larger ranges and more frequent outliers, indicating susceptibility to getting trapped in suboptimal regions. Particularly in F10, BSA demonstrates the tightest interquartile range and minimal number of outliers, reinforcing its robustness in highly multimodal scenarios.

In the hybrid function group (F11–F20), BSA continues to maintain consistent performance across runs, as evidenced by both the low standard deviation and the compact shape of its boxplots. For functions F18 and F19, where optimization complexity increases due to the combination of multiple function modalities, BSA's median and mean values remain very close to the global optimum, while the boxes of other algorithms often display substantial variability, highlighting the superior stability of BSA in handling complex landscapes.

The composition function group (F21–F30) represents the most challenging test cases with high dimensionality and intricate topologies. Remarkably, BSA achieves exceptionally narrow boxes across all these functions, with minimal deviations between the best, median, and worst results. This visually confirms the algorithm's robust convergence behavior and consistent high-quality solutions, corroborating the quantitative superiority observed in Table 1, where BSA attained first rank in all these functions.

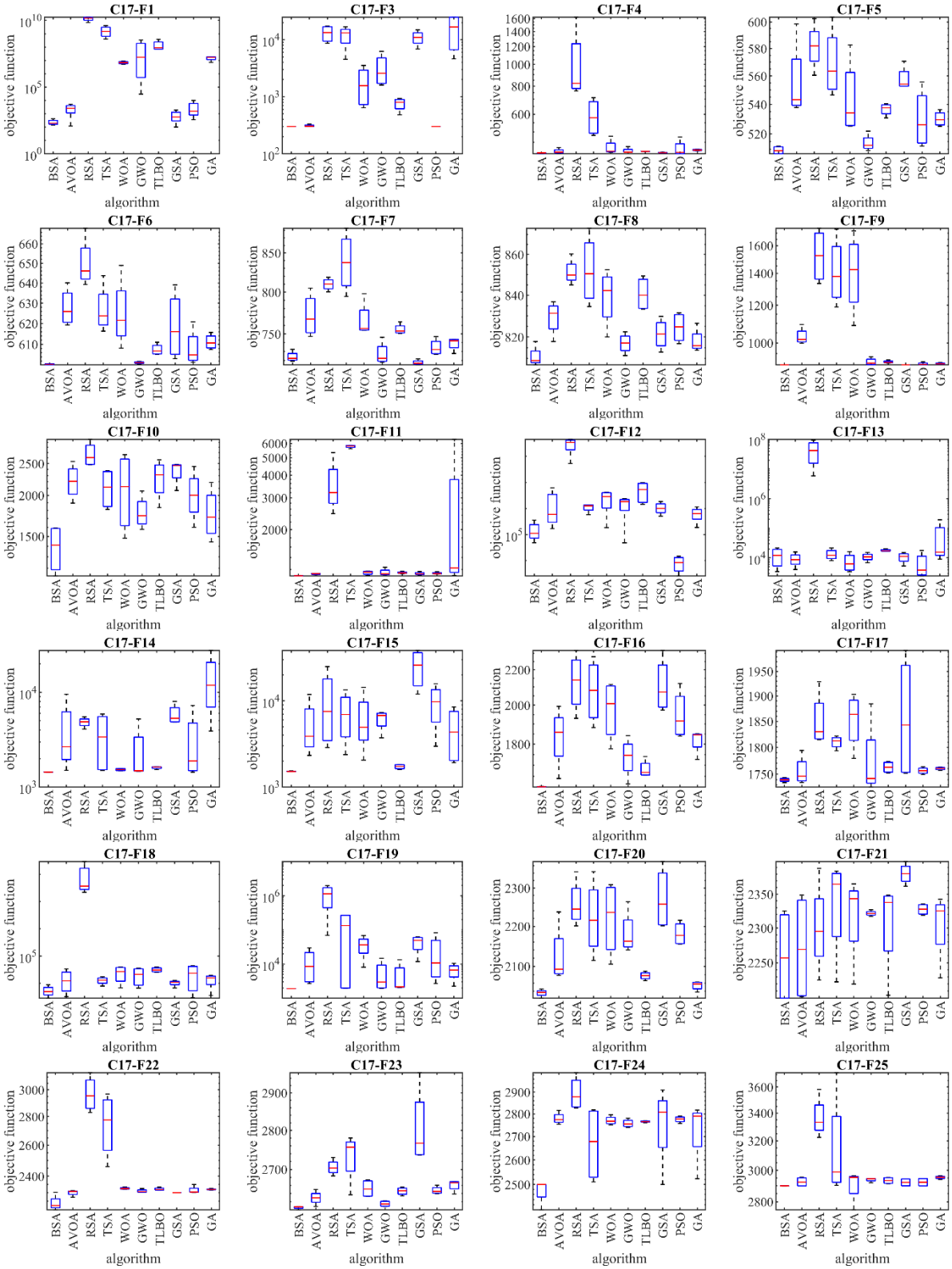
Overall, the boxplot analysis provides compelling qualitative evidence that BSA is not only highly effective but also robust and reliable across diverse problem types. Its ability to consistently produce tight distributions with limited outliers demonstrates a stable balance between exploration and exploitation, ensuring both convergence accuracy and repeatability. These observations reinforce the algorithm's suitability for complex real-world optimization tasks where consistency and predictability are critical.

3.4 Discussion and interpretation

The experimental results clearly demonstrate the

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

Received: July 30, 2025. Revised: November 7, 2025.



 DOI: 10.22266/ijies2026.0131.14

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

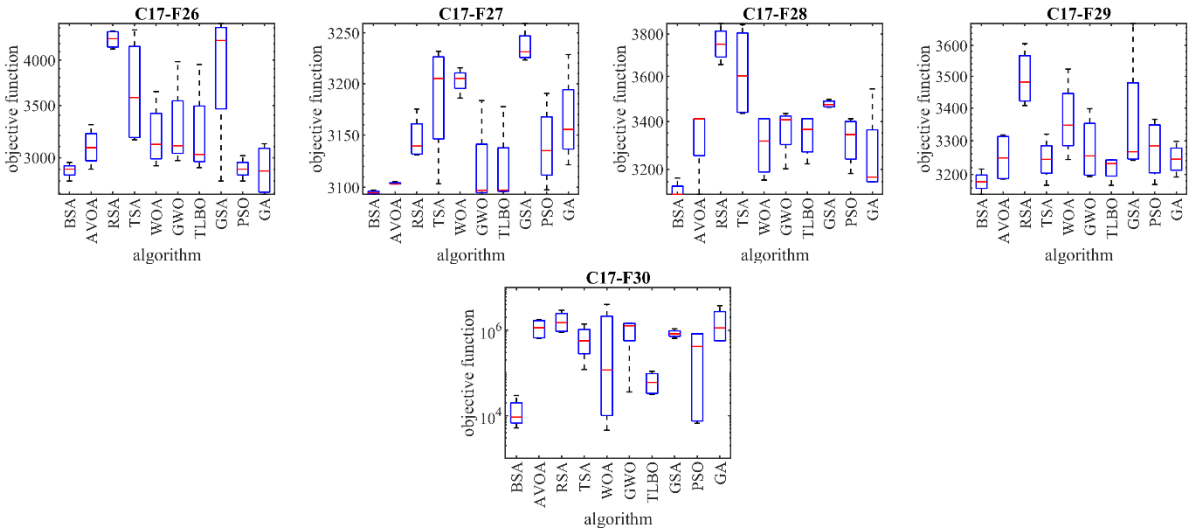


Figure. 1 Boxplot diagrams obtained from BSA and competing algorithms on CEC 2017 test suite

superior performance and robustness of the Bolas Spider Algorithm across diverse optimization landscapes. Out of 29 benchmark functions, BSA achieves the first rank in 24 functions, including F10, F18, F19, and all composition functions F21–F30, reflecting its consistent dominance across unimodal, multimodal, hybrid, and composition problems. This high success rate underscores the algorithm's ability to adaptively balance exploration and exploitation, a crucial property for complex optimization tasks.

The boxplot analysis further highlights BSA's robustness and repeatability. Narrow interquartile ranges and minimal outliers indicate stable convergence, even in highly multimodal or composition landscapes. Functions such as F10, F18, and F19 exemplify the algorithm's capacity to maintain high-quality solutions with low variance, demonstrating effective handling of local optima and complex topologies. Similarly, in composition functions (F21–F30), tight distributions confirm reliable performance under intricate, high-dimensional search spaces.

The Wilcoxon signed-rank test statistically validates these findings, with all p-values below 0.05, confirming that BSA's improvements over nine competing metaheuristics are significant and not due to chance. The algorithm's biologically inspired design underpins its success: pheromone-guided exploration ensures diverse sampling of the solution space, while targeted prey-capture exploitation efficiently drives convergence toward promising regions. This adaptive integration of global and local search mechanisms explains BSA's consistent first-place rankings and narrow result distributions.

In summary, BSA achieves exceptional accuracy, stability, and robustness simultaneously, outperforming contemporary metaheuristics across nearly all benchmark classes. Its performance reflects not only efficient search dynamics but also the effectiveness of its biologically motivated mechanisms in maintaining a reliable balance between diversification and intensification, making it a versatile and trustworthy optimizer for complex continuous problems.

4. Conclusion and future work

This study introduced the Bolas Spider Algorithm (BSA), a novel metaheuristic inspired by the foraging and adaptive behaviors of bolas spiders. The algorithm integrates pheromone-guided exploration with targeted prey-capture exploitation, enabling a dynamic balance between diversification and intensification. This biologically grounded design allows BSA to effectively navigate complex continuous optimization landscapes while avoiding premature convergence and maintaining solution diversity.

Extensive experiments were conducted on the CEC 2017 benchmark suite, comprising 29 continuous functions spanning unimodal, multimodal, hybrid, and composition categories. BSA was compared against nine recent metaheuristics, including GA, PSO, GSA, GWO, TLBO, WOA, TSA, RSA, and AVOA. The results demonstrate that BSA achieves first-rank performance in 24 out of 29 functions, including all composition functions (F21–F30) and critical multimodal/hybrid functions such as F10, F18, and F19. This highlights the algorithm's remarkable adaptability and robustness across diverse problem types.

Qualitative boxplot analyses further confirmed the consistency and stability of BSA, showing narrow

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

interquartile ranges and minimal outliers, even in highly multimodal or high-dimensional landscapes. Statistical validation through the Wilcoxon signed-rank test confirmed that the observed performance gains are significant at the 95% confidence level, emphasizing that BSA consistently outperforms its competitors.

The success of BSA can be attributed to its dual search mechanism: global exploration ensures effective coverage of the solution space, while local exploitation converges efficiently toward promising optima. This adaptive synergy enables BSA to maintain high solution quality while minimizing variance across independent runs, a critical advantage in practical optimization scenarios.

In conclusion, BSA represents a robust, accurate, and statistically validated optimization algorithm capable of tackling a wide range of continuous problems. Its biologically inspired mechanisms, combined with a parameter-efficient design, make it a versatile tool for researchers and practitioners. Future work may explore hybridization with other metaheuristics or applications to real-world engineering and industrial problems, potentially extending BSA's effectiveness to even more challenging and high-dimensional optimization tasks.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, H.Q, S.M, S.A, and Z.M; methodology, M.D, Z.M, H.Q, G.B, and K.E; software, K.E and O.P.M; validation, K.E, M.D, O.P.M, and G.B; formal analysis, Z.M, M.D, K.E, S.A, and O.P.M; investigation, H.Q, G.B, and Z.M; resources, S.M, Z.M, and S.A; data curation, Z.M and K.E; writing—original draft preparation, H.Q, M.D, and S.M; writing—review and editing, Z.M, G.B, and K.E; visualization, K.E and O.P.M; supervision, M.D; project administration, K.E, S.M, O.P.M, S.A, and H.Q; funding acquisition, K.E.

References

[1] A. M. Zalzala, R. K. Jawad, A. A. M. M. AL-Salih, M. A. Ahmed, and I. K. Ibraheem, "Application of Spider-Tailed Horned Viper Optimization for Unit Commitment Problem in Power Systems", *INASS Express*, Vol. 1, No. 1, pp. 10-18, 2025, doi: 10.22266/inassexpress.2025.002.

[2] F. Werner, B. Batiha, T. Hamadneh, G. M. Gharib, W. Aribowo, and M. Dehghani, "Application of the Builder Optimization Algorithm for Sustainable Lot Size Optimization in Supply Chain Management: A Comprehensive Analysis and Comparison with Metaheuristic Approaches", *INASS Express*, Vol. 1, No. 1, pp. 39-50, 2025, doi: 10.22266/inassexpress.2025.005.

[3] Z. Monrazeri, O. P. Malik, and M. Dehghani, "Capacitor Placement in Power Distribution Networks for Voltage Profile Improvement and Loss Reduction Using Revolution Optimization Algorithm", *INASS Express*, Vol. 1, No. 1, pp. 29-38, 2025, doi: 10.22266/inassexpress.2025.004.

[4] T. Hamadneh, B. Batiha, G. M. Gharib, and W. Aribowo, "Application of Orangutan Optimization Algorithm for Feature Selection Problems", *INASS Express*, Vol. 1, No. 1, pp. 1-9, 2025, doi: 10.22266/inassexpress.2025.001.

[5] T. Hamadneh, B. Batiha, and G. M. Gharib, "Enhanced Distributed Generation Placement Using Carpet Weaver Optimization: A Comparative Metaheuristic Approach", *INASS Express*, Vol. 1, No. 1, pp. 19-28, 2025, doi: 10.22266/inassexpress.2025.003.

[6] D. Das, A. S. Sadiq, and S. Mirjalili, "Optimization Methods: Deterministic Versus Stochastic", *Optimization Algorithms in Machine Learning: A Meta-heuristics Perspective*. Singapore: Springer Nature Singapore, pp. 13-19, 2025.

[7] R. Abu-Gdairi, R. Mareay, and M. Badr, "On Multi-Granulation Rough Sets with Its Applications", *Computers, Materials & Continua*, Vol. 79, No. 1, pp. 1025-1038, 2024.

[8] H. Qawaqneh, "New contraction embedded with simulation function and cyclic (α, β)-admissible in metric-like spaces", *International Journal of Mathematics and Computer Science*, Vol. 15, No. 4, pp. 1029-1044, 2020.

[9] T. Hamadneh, N. Athanasopoulos, and M. Ali, "Minimization and positivity of the tensorial rational Bernstein form", In: *Proc. of 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT - Proceedings)*, pp. 474-479, 2019, doi: 8717503.

[10] T. Hamadneh and R. Wisniewski, "The Barycentric Bernstein Form for Control Design", In: *Proc. of 2018 Annual American Control Conference (ACC)*, pp. 3738-3743, 2018, doi: 10.23919/ACC.2018.8431599.

[11] M. Dehghani, Z. Montazeri, E. Trojovská, and P. Trojovský, "Coati Optimization Algorithm: A new bio-inspired metaheuristic algorithm for

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

solving optimization problems", *Knowledge-Based Systems*, Vol. 259, p. 110011, 2023, doi: 10.1016/j.knosys.2022.110011.

[12] F. W. Glover and G. A. Kochenberger, *Handbook of metaheuristics*, Springer Science & Business Media, 2003.

[13] V. C. S. S and A. H. S, "Nature inspired meta heuristic algorithms for optimization problems", *Computing*, Vol. 104, No. 2, pp. 251-269, 2022, doi: 10.1007/s00607-021-00955-5.

[14] J. J. Lian *et al.*, "Trend-Aware Mechanism for Metaheuristic Algorithms", *Applied Soft Computing*, Vol. 182, p. 113505, 2025, doi: 10.1016/j.asoc.2025.113505.

[15] P. Trojovský and M. Dehghani, "A new bio-inspired metaheuristic algorithm for solving optimization problems based on walruses behavior", *Scientific Reports*, Vol. 13, No. 1, p. 8775, 2023, doi: 10.1038/s41598-023-35863-5.

[16] Z. Wang *et al.*, "An adaptive equilibrium optimizer with information enhancement for global optimization", *Cluster Computing*, Vol. 28, No. 10, p. 673, 2025, doi: 10.1007/s10586-025-05334-9.

[17] K. Khlie, A. Pugalenthi, Z. Benmamoun, W. Aribowo, and M. Dehghani, "Sustainable Supply Chain Optimization: A Breakthrough in Swarm-based Artificial Intelligence", *Engineering, Technology & Applied Science Research*, Vol. 15, No. 3, pp. 23125-23132, 2025, doi: 10.48084/etasr.10505.

[18] M. Q. Ibrahim, N. K. Hussein, D. Guinovart, and M. Qaraad, "Optimizing Convolutional Neural Networks: A Comprehensive Review of Hyperparameter Tuning Through Metaheuristic Algorithms", *Archives of Computational Methods in Engineering*, 2025, doi: 10.1007/s11831-025-10292-x.

[19] J. Kennedy and R. Eberhart, "Particle swarm optimization", In: *Proc. of ICNN'95 - International Conference on Neural Networks*, Vol. 4, pp. 1942-1948, 1995, doi: 10.1109/ICNN.1995.488968.

[20] J. H. Holland, "Genetic Algorithms", *Scientific American*, Vol. 267, No. 1, pp. 66-73, 1992.

[21] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization", *IEEE Computational Intelligence Magazine*, Vol. 1, No. 4, pp. 28-39, 2006, doi: 10.1109/MCI.2006.329691.

[22] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing", *Science*, Vol. 220, No. 4598, pp. 671-680, 1983.

[23] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer", *Advances in Engineering Software*, Vol. 69, pp. 46-61, 2014.

[24] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems", *Computer-Aided Design*, Vol. 43, No. 3, pp. 303-315, 2011, doi: 10.1016/j.cad.2010.12.015.

[25] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A Gravitational Search Algorithm", *Information Sciences*, Vol. 179, No. 13, pp. 2232-2248, 2009, doi: 10.1016/j.ins.2009.03.004.

[26] I. Matoušová, P. Trojovský, M. Dehghani, E. Trojovská, and J. Kostra, "Mother optimization algorithm: a new human-based metaheuristic approach for solving engineering optimization", *Scientific Reports*, Vol. 13, No. 1, p. 10312, 2023, doi: 10.1038/s41598-023-37537-8.

[27] Y. Deng, Y. Jiang, S. Zheng, and J. Ma, "Adams-Bashforth-Moulton optimizer: a novel metaheuristic algorithm for solving engineering optimization problems", *Cluster Computing*, Vol. 28, No. 12, p. 766, 2025, doi: 10.1007/s10586-025-05508-5.

[28] J. Wang and Z. Shang, "Traffic jam optimizer: A novel swarm-based metaheuristic algorithm for solving global optimization problems", *Applied Mathematical Modelling*, Vol. 150, p. 116410, 2026, doi: 10.1016/j.apm.2025.116410.

[29] M. Song, J. Lin, X. Liu, H. Jia, and S. Luo, "Octopus optimization algorithm: a novel single- and multi-objective optimization algorithm for optimization problems", *Cluster Computing*, Vol. 28, No. 8, p. 484, 2025, doi: 10.1007/s10586-025-05141-2.

[30] Y. Xiao, H. Cui, R. A. Khurma, and P. A. Castillo, "Artificial lemming algorithm: a novel bionic meta-heuristic technique for solving real-world engineering optimization problems", *Artificial Intelligence Review*, Vol. 58, No. 3, p. 84, 2025, doi: 10.1007/s10462-024-11023-7.

[31] T. M. Shami, D. Grace, A. Burr, and P. D. Mitchell, "Single candidate optimizer: a novel optimization algorithm", *Evolutionary Intelligence*, Vol. 17, No. 2, pp. 863-887, 2024, doi: 10.1007/s12065-022-00762-7.

[32] M. Abdel-Basset, R. Mohamed, and M. Abouhawwash, "Crested Porcupine Optimizer: A new nature-inspired metaheuristic", *Knowledge-Based Systems*, Vol. 284, p. 111257, 2024, doi: 10.1016/j.knosys.2023.111257.

[33] F. A. Hashim and A. G. Hussien, "Snake Optimizer: A novel meta-heuristic optimization algorithm", *Knowledge-Based Systems*, Vol. 242, p. 108320, 2022, doi: 10.1016/j.knosys.2022.108320.

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/

[34] L. Wang, Q. Cao, Z. Zhang, S. Mirjalili, and W. Zhao, "Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems", *Engineering Applications of Artificial Intelligence*, Vol. 114, p. 105082, 2022, doi: 10.1016/j.engappai.2022.105082.

[35] T. Hamadneh *et al.*, "Rabbit and Turtle Algorithm: A Novel Metaheuristic for Solving Complex Engineering Optimization Problems", *International Journal of Intelligent Engineering & Systems*, Vol. 18, No. 6, pp. 426-438, 2025, doi: 10.22266/ijies2025.0731.27.

[36] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization", *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, pp. 67-82, 1997.

[37] N. Awad, M. Ali, J. Liang, B. Qu, P. Suganthan, and P. Definitions, "Evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization", *Technology Report*, 2016.

[38] F. Wilcoxon, "Individual Comparisons by Ranking Methods", *Biometrics Bulletin*, Vol. 1, No. 6, pp. 80-83, 1945, doi: 10.2307/3001968.

[39] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm", *Advances in Engineering Software*, Vol. 95, pp. 51-67, 2016, doi: 10.1016/j.advengsoft.2016.01.008.

[40] S. Kaur, L. K. Awasthi, A. L. Sangal, and G. Dhiman, "Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization", *Engineering Applications of Artificial Intelligence*, Vol. 90, p. 103541, 2020, doi: 10.1016/j.engappai.2020.103541.

[41] L. Abualigah, M. A. Elaziz, P. Sumari, Z. W. Geem, and A. H. Gandomi, "Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer", *Expert Systems with Applications*, Vol. 191, p. 116158, 2022, doi: 10.1016/j.eswa.2021.116158.

[42] B. Abdollahzadeh, F. S. Gharehchopogh, and S. Mirjalili, "African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems", *Computers & Industrial Engineering*, Vol. 158, p. 107408, 2021, doi: 10.1016/j.cie.2021.107408.

This article is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
License details: https://creativecommons.org/licenses/by-sa/4.0/