Iranian Journal of Fuzzy Systems

Volume 22, Number 6, (2025), pp. 125-145

Original paper

University of Sistan and Baluchestan

Adaptive fuzzy swarm-based search algorithm (AFSSA) for complex engineering optimization

R. Etesami [1], M. Madadi [2] and F. Keynia [3]

[1,2] *Department of Statistics, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran.*
[3] *Department of Energy Management and Optimization, Institute of Science and High Technology and Environmental Sciences, Graduate University of Advanced Technology, Kerman, Iran.*

Rezaetesamii@gmail.com, madadi@uk.ac.ir, Fkeynia@gmail.com

Abstract

In recent years, swarm intelligence metaheuristic algorithms have emerged as powerful tools for solving real-world engineering optimization problems. However, their performance often degrades when applied to complex, high-dimensional problems. To address this limitation, we propose an Adaptive Fuzzy Swarm-based Search Algorithm (AFSSA), which incorporates a Fuzzy Dynamic Control Mechanism to dynamically adjust the optimization coefficients of swarm intelligence algorithms. AFSSA employs a Mamdani fuzzy inference system to enable smooth phase transitions during optimization, ensuring adaptability to the problem's unique characteristics. In this study, AFSSA is applied to enhance the acceleration coefficients of Particle Swarm Optimization (PSO) and Golden Search Optimization (GSO), resulting in AFSSA-PSO and AFSSA-GSO. The performance of these modified algorithms is evaluated on 23 standard benchmark functions (with dimensions of 30, 100, and 500) and the CEC2019 test suite, showing competitive results compared to other well-known optimization methods. Additionally, AFSSA is tested on data clustering problems, further demonstrating its versatility in handling complex real-world applications.

Keywords: Adaptive fuzzy, swarm intelligence, complex optimization problems, high-dimensional optimization problems.

1 Introduction

Metaheuristic algorithms, inspired by natural phenomena, are effective in solving complex optimization problems across fields like engineering and data science [7, 8, 9, 10, 19]. Among them, swarm intelligence techniques such as PSO [5] and ACO [4] has gained popularity due to its ability to explore solution spaces and perform global searches efficiently [15]. However, their performance tends to decline in high-dimensional or complex problem spaces [24], mainly due to difficulties in balancing exploration and exploitation [14].

In swarm-based algorithms, this balance is typically controlled by specific coefficients, which are often fixed or randomly set. These static parameters may not adapt well to the dynamic nature of real-world optimization problems, resulting in premature convergence or suboptimal solutions. Consequently, there has been growing interest in adaptive strategies that dynamically adjust these parameters during the optimization process to enhance algorithm performance [13].

Fuzzy logic-based methods, despite their relatively limited popularity in the data mining field, offer significant advantages due to their capacity for linguistic summarization and approximate reasoning. The ability of fuzzy systems to model uncertainty and vagueness through human-readable linguistic rules enables better interpretability and dynamic adaptability. This capability supports nuanced control over optimization parameters by capturing the evolving complexity of the search landscape in a way that conventional crisp methods cannot.

Corresponding Author: M. Madadi
Received: May 2025; Revised: September 2025; Accepted: November 2025.
https://doi.org/10.22111/ijfs.2025.52217.9209

To address these challenges, this paper proposes the Adaptive Fuzzy Swarm-based Search Algorithm (AFSSA), which utilizes a Mamdani fuzzy inference system to intelligently and dynamically tune algorithm coefficients in real time. AFSSA is integrated with both the well-established PSO and the more recent GSO algorithm [25], each of which suffers from limitations due to non-adaptive parameters. By leveraging fuzzy linguistic summarization, AFSSA facilitates smooth phase transitions and enhances convergence behavior, resulting in significantly improved performance for complex and high-dimensional optimization problems.

Contributions

The main contributions of this work are:

1. A novel Fuzzy Dynamic Control Mechanism using Mamdani inference to adaptively adjust swarm optimization parameters, enabling smooth phase transitions during search.
2. Development of AFSSA-PSO and AFSSA-GSO variants showing competitive performance on 23 benchmark functions (up to 500D) and the CEC2019 test suite.
3. Successful application to data clustering problems, demonstrating the method's versatility beyond numerical optimization.

The paper is organized as follows: Section 2 reviews related works. Section 3 presents the proposed AFSSA algorithm, which utilizes a Mamdani fuzzy inference system to dynamically adjust the optimization coefficients. Section 4 discusses the results, and Section 5 concludes the paper by summarizing the key findings.

Table 1: Coefficients controlling exploration and exploitation in widely used swarm intelligence algorithms

Algorithm	**Coefficient**	**Related to Exploration and Exploitation**
Particle Swarm Optimization (PSO) [5]	c_1	Exploration (Cognitive coefficient)
	c_2	Exploitation (Social coefficient)
Ant Colony Optimization (ACO)[4]	α	Exploitation (Pheromone importance)
	β	Exploration (Distance or cost importance)
Artificial Bee Colony (ABC)[15]	ϕ	Both (Balance between exploration and exploitation)
Gravitational Search Algorithm (GSA)[27]	A	Both (Adjusts intensity of gravitational force)
Differential Evolution (DE)[30]	F	Exploration (Scaling factor)
	CR	Exploitation (Crossover rate)
Bat Algorithm (BA) [35]	A	Exploration (Amplitude)
	r	Exploitation (Rate of emission)
Cuckoo Search Algorithm (CSA)[36]	p	Exploration (Egg-laying probability)
	L	Exploitation (Levy flight factor)
Harmony Search Algorithm (HSA)[12]	$HMCR$	Exploitation (Harmony Memory Considering Rate)
	PAR	Exploration (Pitch Adjusting Rate)
Shuffled Frog Leaping Algorithm (SFLA)[11]	r_{local}	Exploitation (Local search factor)
	r_{global}	Exploration (Global search factor)

2 Related works

Swarm intelligence algorithms, particularly Particle Swarm Optimization (PSO) and other metaheuristic methods, have attracted substantial attention for solving complex real-world engineering problems. Despite their popularity, traditional swarm algorithms often suffer from premature convergence, lack of adaptability, and difficulty in balancing exploration and exploitation, especially in high-dimensional and nonlinear optimization landscapes. To address these challenges, several studies have focused on integrating fuzzy logic with swarm intelligence to develop adaptive and dynamic control mechanisms.

Early work by Shi and Eberhart introduced fuzzy systems to dynamically adjust PSO parameters, such as inertia weight, aiming at improved convergence and avoidance of local optima [29]. Similarly, Aine et al. proposed an adaptive evolutionary framework that dynamically tunes optimization parameters, enhancing the algorithm's flexibility in diverse problem landscapes [3]. Building on these foundations, Xu developed an adaptive PSO that modifies acceleration coefficients on-the-fly to prevent premature convergence and enhance global search capability [34]. Juang et al. further extended this concept by proposing a fuzzy adaptive PSO (AFPSO), which significantly improved search efficiency and accuracy in multimodal problems [17].

Expanding these hybrid adaptive strategies, Taieb et al. presented a novel Adaptive Takagi–Sugeno fuzzy Model Predictive Control is designed using a hybrid PSO-Cuckoo Search (PSOCS) algorithm [31]. By combining the global search ability of PSO with the Lévy flight and elimination mechanisms of Cuckoo Search, PSOCS adaptively optimizes fuzzy controller parameters for nonlinear constrained systems, showing robust control performance and better stability over standalone PSO or CS methods.

Further innovations include the use of neuro-fuzzy inference systems to improve accuracy and robustness in control tasks. Wang et al. proposed a model-free Neuro-Fuzzy Q-Learning (EGQL) algorithm that integrates the strength of adaptive neuro-fuzzy inference systems (ANFIS) with evolutionary search techniques, effectively overcoming gradient-based limitations and achieving precise control in uncertain nonlinear systems [33].

In the field of high-dimensional optimization, Abiyev and Tunay introduced the Hypercube Optimization (HO) algorithm, a stochastic search method that encapsulates solutions within a hypercube search space, optimizing it via displacement and shrinking strategies [2]. Tunay and Abiyev further improved this algorithm (HOS+) by incorporating random perturbations to prevent premature convergence and trapping in local minima, showing superior performance on several benchmark functions with dimensions up to several thousand [32].

Fuzzy logic's adaptivity has also been effectively combined with metaheuristic algorithms for fault-tolerant control in nonlinear systems. Patel and Shah developed fuzzy logic-based harmonic search algorithms, employing interval type-2 fuzzy systems to dynamically adapt parameters and handle uncertainty, greatly enhancing fault-tolerant control performance under actuator faults [26]. Similarly, Mahmoudi et al. hybridized neuro-fuzzy models with multiple metaheuristic algorithms—such as the Firefly Algorithm (FA), Grey Wolf Optimization (GWO), and Bees Algorithm (BA)—for accurate soil moisture content estimation, demonstrating improved predictive accuracy across different climatic zones [20].

Energy management systems and renewable energy applications have also benefited from fuzzy-swarm hybrid approaches. Ibrahim et al. introduced a Fuzzy Logic-Based PSO (FLB-PSO) tailored for integrated energy management considering battery degradation, showing faster convergence, cost savings, and enhanced robustness over classical PSO [16]. Moreover, hybrid fuzzy logic-proportional integral (FLC-PI) controllers optimized via metaheuristic algorithms such as Grey Wolf Optimization and Harris Hawks Optimization, were successfully applied to enhance power quality and voltage stability in high-penetration grid-connected photovoltaic systems [6].

Collectively, these advances underline the pivotal role of adaptive fuzzy mechanisms in enhancing swarm intelligence optimizers across diverse application domains. By embedding fuzzy inference systems—particularly of the Mamdani and Takagi–Sugeno types—as dynamic control units, these hybrid approaches modulate algorithm parameters such as acceleration coefficients, inertia weights, population sizes, and mutation rates in response to evolving search conditions. This phase-shift capability facilitates smooth transitions between exploration and exploitation stages, allowing the optimizer to self-tune to problem-specific features, thus overcoming the conventional limitations of static parameters settings.

Nevertheless, most of the existing methods primarily focus on low- to medium-dimensional problems or specific application domains, often lacking a systematic evaluation on large-scale benchmarks and highly complex optimization tasks. In contrast, the proposed Adaptive Fuzzy Swarm-based Search Algorithm (AFSSA) explicitly addresses these gaps by incorporating a Mamdani fuzzy dynamic control mechanism that adapts swarm coefficients in real time. As demonstrated in this study, AFSSA has been rigorously evaluated not only on 23 standard benchmark functions, but also on challenging large-scale CEC2019 functions with dimensions up to 500. Furthermore, AFSSA has been successfully applied to complex real-world problems such as data clustering, where it consistently achieves superior performance compared to classical and hybrid metaheuristics. These results highlight the effectiveness and robustness of AFSSA in handling both high-dimensional search spaces and complex problem structures, establishing it as a state-of-the-art adaptive swarm-based optimizer.

3 Proposed AFSSA algorithm

In this paper, the Mamdani fuzzy inference system is employed as a dynamic control mechanism for adjusting optimization coefficients. The Mamdani system incorporates two primary inputs: the first input (curve density ratio), which serves as an indicator of search space variability, and the second input (distribution type), which defines the sample distribution pattern. The output of this fuzzy system consists of the coefficients c_1 and c_2, which are adaptively adjusted based on these input conditions to maintain an optimal balance between exploration and exploitation. This dynamic adjustment allows the algorithm to choose the most suitable search strategy according to the real-time conditions in the search space. The steps of the AFSSA algorithm are outlined as follows:

Step 1: Initialization Phase

Population Initialization: AFSSA begins by initializing a population of N candidate solutions, each representing a point in the search space. The position of the i-th candidate solution, O_i, is initialized as:

$$O_i = \mathrm{lb} + \mathrm{rand} \times (\mathrm{ub} - \mathrm{lb}), \quad i = 1, 2, \dots, N,$$

where *lb* and *ub* represent the lower and upper bounds of the search space, respectively. Let O_{ij} denote the position of particle i in dimension j, and N refers to the total number of particles in the population.

Calculate the Mean Position in Dimension j: First, the mean position of particles in dimension j is calculated as follows:

$$\bar{O}_j = \frac{1}{N} \sum_{i=1}^{N} O_{ij}.$$

Calculate the Standard Deviation in Dimension j: Then, the standard deviation of particle positions in dimension j, denoted by D_t, is calculated as follows:

$$D_t = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (O_{ij} - \bar{O}_j)^2},$$

where: $\bar{O}_j$ represents the mean position of particles in dimension j. D_t is the standard deviation of particle positions in dimension j, which measures the spread of positions around the mean in that dimension.

Fitness Evaluation: Each candidate's quality is evaluated using the fitness function $f(O_i)$. The candidate with the best fitness score in the initial population is designated as O_{gbest}.

Step 2: Choosing the best distribution at time t:

To evaluate the state of the search space at time t, random samples are generated from three probability distributions centered around O_{gbest}. These distributions are defined as follows:

Triangular Distribution Sampling

The Triangular distribution centers the particles closely around the current best particle O_{gbest}, focusing on minimal spread and thus supporting fine-tuned local search. This distribution's probability density function (PDF) is defined as follows:

$$f(x; a, b, c) = \begin{cases} \frac{2(x-a)}{(b-a)(c-a)}, & \text{for } a \le x < c \\ \frac{2(b-x)}{(b-a)(b-c)}, & \text{for } c \le x \le b \\ 0, & \text{otherwise} \end{cases}$$

where: $a = O_{\mathrm{gbest}} - \frac{D_t}{2}$ is the lower bound, $b = O_{\mathrm{gbest}} + \frac{D_t}{2}$ is the upper bound, $c = O_{\mathrm{gbest}}$ represents the mode, or peak.

This distribution restricts particles within a narrow band around O_{gbest}, making it effective for exploitation-focused local searches.

Gaussian Distribution Sampling

The Gaussian distribution enables moderate particle spread around O_{gbest}, allowing a balanced search between local exploitation and broader exploration. For each dimension j, the Gaussian distribution is centered at O_{gbest} with standard deviation $\sigma = \frac{D_t}{2}$, and its PDF is given by:

$$f(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

where: $\mu = O_{\mathrm{gbest}}$ is the mean, $\sigma = \frac{D_t}{2}$ defines the spread.

The bell-shaped curve of the Gaussian distribution provides flexibility, allowing particles to explore the vicinity of O_{gbest} more moderately,which aids in maintaining a balance between the algorithm's exploration and exploitation.

Trapezoidal Distribution Sampling

The Trapezoidal distribution produces the widest spread of particles, promoting a more global search around O_{gbest}. For each dimension, the Trapezoidal distribution's PDF is structured as follows:

$$f(x;a,b,c,d) = \begin{cases} \frac{2(x-a)}{(b-a)(d-a)}, & \text{for } a \leq x < b \\ \frac{2}{d-a}, & \text{for } b \leq x \leq c \\ \frac{2(d-x)}{(d-c)(d-a)}, & \text{for } c < x \leq d \\ 0, & \text{otherwise} \end{cases}$$

where: $a = O_{\text{gbest}} - D_t$, $b = O_{\text{gbest}} - \frac{D_t}{2}$, $c = O_{\text{gbest}} + \frac{D_t}{2}$, $d = O_{\text{gbest}} + D_t$.

This distribution spans a broader range in the search space, making it suitable for exploring diverse areas and encouraging a more global search behavior.

Each distribution in the AFSSA framework is selected based on the position and fitness of the best particle, O_{gbest}, relative to the farthest particle, with D_t guiding the boundaries and spread. This multi-distribution sampling approach enables the AFSSA algorithm to balance global exploration and local exploitation dynamically, improving its optimization efficacy.

Step 3: Curve Variation Density Analysis

Curve Variation Density Analysis characterizes the fitness landscape by computing the Curve Density Ratio s_{ratio}, defined as the ratio of the standard deviation of the distance between the best and farthest particle X_{id} to the standard deviation of their fitness values:

$$s_{\text{ratio}} = \frac{\text{sd}(X_{id})}{\text{sd}(\text{fitness})}.$$

Interpretation of s_{ratio}:

- **High s_{ratio}**: Indicates a smooth landscape with minimal fitness change across space, favoring global exploration.
- **Moderate s_{ratio}**: Suggests a balanced landscape, requiring a trade-off between exploration and exploitation.
- **Low s_{ratio}**: Implies a steep landscape where small movements yield significant fitness shifts, favoring local exploitation.

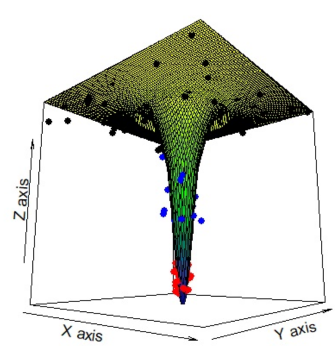

Figure 1: Particle behavior based on Curve Variation Density Analysis

The Curve Variation Density Analysis (see Figure 1) reveals that particles adapt their movement based on the curve density ratio s_{ratio} of their respective regions in the search space. Black particles, located in smooth regions with high s_{ratio}, are designed for rapid movement to perform extensive global exploration. In contrast, blue particles operate in moderately complex landscapes, adopting a balanced movement strategy that utilizes both their own historical data and attraction to the global best position, thus contributing to both exploration and exploitation.

Red particles are found in rugged regions with low s_{ratio}, where even minor positional adjustments significantly affect fitness. As such, they emphasize fine-grained local searches around their own and their neighborhood's best positions. This targeted movement supports refined optimization in promising areas.

Step 4: Fuzzy Membership Functions

In this section, the fuzzy membership functions for the input variable curve variation density and the output variables $C1$ and $C2$ are introduced.

Fuzzy Membership Functions for s_{ratio}

$$\mu_{\text{Low}}(s_{\text{ratio}}) = \begin{cases} 1, & s_{\text{ratio}} \leq 0.5 \\ \frac{1-s_{\text{ratio}}}{0.5}, & 0.5 < s_{\text{ratio}} < 1 \\ 0, & s_{\text{ratio}} \geq 1 \end{cases}$$

$$\mu_{\text{Moderate}}(s_{\text{ratio}}) = \begin{cases} 0, & s_{\text{ratio}} \leq 0.5 \\ \frac{s_{\text{ratio}}-0.5}{0.5}, & 0.5 < s_{\text{ratio}} < 1 \\ \frac{1.5-s_{\text{ratio}}}{0.5}, & 1 < s_{\text{ratio}} < 1.5 \\ 0, & s_{\text{ratio}} \geq 1.5 \end{cases}$$

$$\mu_{\text{High}}(s_{\text{ratio}}) = \begin{cases} 0, & s_{\text{ratio}} \leq 1 \\ \frac{s_{\text{ratio}}-1}{0.5}, & 1 < s_{\text{ratio}} < 1.5 \\ 1, & s_{\text{ratio}} \geq 1.5 \end{cases}$$

Fuzzy Membership Functions for Coefficients c_1 and c_2

The coefficients c_1 and c_2 are modeled using fuzzy membership functions, defined by linguistic terms ranging from *Extremely Low* to *Extremely High*. The triangular membership functions are given as:

$$\mu_{\text{EL}}(x) = \begin{cases} \frac{x}{0.2}, & 0 \leq x < 0.2 \\ 0, & \text{otherwise} \end{cases}$$

$$\mu_{\text{VL}}(x) = \begin{cases} 0, & x < 0.1 \text{ or } x > 0.3 \\ \frac{x-0.1}{0.2}, & 0.1 \leq x < 0.2 \\ \frac{0.3-x}{0.1}, & 0.2 \leq x < 0.3 \end{cases}$$

$$\mu_{\text{L}}(x) = \begin{cases} 0, & x < 0.2 \text{ or } x > 0.4 \\ \frac{x-0.2}{0.2}, & 0.2 \leq x < 0.3 \\ \frac{0.4-x}{0.1}, & 0.3 \leq x < 0.4 \end{cases}$$

$$\mu_{\text{ML}}(x) = \begin{cases} 0, & x < 0.3 \text{ or } x > 0.5 \\ \frac{x-0.3}{0.2}, & 0.3 \leq x < 0.4 \\ \frac{0.5-x}{0.1}, & 0.4 \leq x < 0.5 \end{cases}$$

$$\mu_{\text{M}}(x) = \begin{cases} 0, & x < 0.4 \text{ or } x > 0.6 \\ \frac{x-0.4}{0.2}, & 0.4 \leq x < 0.5 \\ \frac{0.6-x}{0.1}, & 0.5 \leq x < 0.6 \end{cases}$$

$$\mu_{\text{MH}}(x) = \begin{cases} 0, & x < 0.5 \text{ or } x > 0.7 \\ \frac{x-0.5}{0.2}, & 0.5 \leq x < 0.6 \\ \frac{0.7-x}{0.1}, & 0.6 \leq x < 0.7 \end{cases}$$

$$\mu_{\text{H}}(x) = \begin{cases} 0, & x < 0.6 \text{ or } x > 0.8 \\ \frac{x-0.6}{0.2}, & 0.6 \leq x < 0.7 \\ \frac{0.8-x}{0.1}, & 0.7 \leq x < 0.8 \end{cases}$$

$$\mu_{\text{VH}}(x) = \begin{cases} 0, & x < 0.7 \text{ or } x > 0.9 \\ \frac{x-0.7}{0.2}, & 0.7 \leq x < 0.8 \\ \frac{0.9-x}{0.1}, & 0.8 \leq x < 0.9 \end{cases}$$

$$\mu_{\text{EH}}(x) = \begin{cases} 0, & x < 0.8 \\ \frac{x-0.8}{0.2}, & 0.8 \leq x < 1 \\ 1, & x = 1 \end{cases}$$

These membership functions provide a fuzzy representation of the adaptive coefficients c_1 and c_2. This formulation allows the optimization algorithm to dynamically adjust their values in response to the search space characteristics.

Step 5: Adaptive Fuzzy Rules for Coefficient Adjustment

The coefficients c_1 and c_2 are adaptively tuned based on the slope ratio *s_ratio* (Low, Moderate, High) and the selected distribution type (Triangular, Gaussian, Trapezoidal). Table 2 summarizes the fuzzy rules, which balance exploration and exploitation.

Table 2: Adaptive Fuzzy Rules for Coefficients c_1 and c_2

s_ratio	**Distribution**	c_1	c_2	**Focus**
High	Triangular	$\mu_{\mathrm{EH}}(x)$	$\mu_{\mathrm{EL}}(x)$	Strong exploitation
High	Gaussian	$\mu_{\mathrm{VH}}(x)$	$\mu_{\mathrm{VL}}(x)$	Exploitation-biased
High	Trapezoidal	$\mu_{\mathrm{H}}(x)$	$\mu_{\mathrm{L}}(x)$	Local exploitation
Moderate	Triangular	$\mu_{\mathrm{MH}}(x)$	$\mu_{\mathrm{ML}}(x)$	Exploitation-skewed
Moderate	Gaussian	$\mu_{\mathrm{M}}(x)$	$\mu_{\mathrm{M}}(x)$	Balanced
Moderate	Trapezoidal	$\mu_{\mathrm{ML}}(x)$	$\mu_{\mathrm{MH}}(x)$	Mild exploration
Low	Triangular	$\mu_{\mathrm{L}}(x)$	$\mu_{\mathrm{H}}(x)$	Exploration-biased
Low	Gaussian	$\mu_{\mathrm{VL}}(x)$	$\mu_{\mathrm{VH}}(x)$	Strong exploration
Low	Trapezoidal	$\mu_{\mathrm{EL}}(x)$	$\mu_{\mathrm{EH}}(x)$	Maximum exploration

These rules ensure that c_1 and c_2 adapt effectively, allowing the algorithm to align its search strategy with the complexity of the optimization landscape and to maintain a balance between local exploitation and global exploration.

Algorithm 1 Pseudo-code of the Adaptive Fuzzy Swarm-based Search Algorithm (AFSSA)

Initialize population of N candidate solutions O_i within bounds $[lb, ub]$
for each particle $i = 1, \dots, N$ **do**
  for each dimension $j = 1, \dots, d$ **do**
    $O_{ij} \leftarrow lb_j + rand() \times (ub_j - lb_j)$
  end for
end for
Evaluate fitness $f(O_i)$ for each particle
$O_{\mathrm{gbest}} \leftarrow$ best particle based on $f(O_i)$
for iteration $t = 1$ to $MaxIter$ **do**
  Compute mean position $\bar{O}_j$ and standard deviation D_t for each dimension
  Select best distribution (Triangular, Gaussian, Trapezoidal) centered at O_{gbest} according to spread D_t
  for each particle i **do**
    Sample candidate positions $\hat{O}_{ij}$ from chosen distribution
    Compute curve density ratio $s_{\mathrm{ratio}} \leftarrow \frac{sd(X_{id})}{sd(\mathrm{fitness})}$
    Fuzzify s_{ratio} and distribution type using predefined membership functions
    Apply fuzzy inference rules (Table 2) to adapt coefficients c_1, c_2
    Update particle position (PSO):

$$O_{ij}^{t+1} \leftarrow c_1 \cdot rand_1 \cdot (O_{\mathrm{pbest},ij} - O_{ij}^{t}) + c_2 \cdot rand_2 \cdot (O_{\mathrm{gbest},j} - O_{ij}^{t}) + \hat{O}_{ij}$$

    Boundary check: ensure $O_{ij}^{t+1} \in [lb_j, ub_j]$
    Fitness evaluation: $f(O_i^{t+1})$
    if $f(O_i^{t+1})$ is better than $f(O_{\mathrm{pbest},i})$ **then**
      $O_{\mathrm{pbest},i} \leftarrow O_i^{t+1}$
    end if
  end for
  Update O_{gbest} if any O_i^{t+1} has better fitness
end for
Return O_{gbest} as the best obtained solution

3.1 Exploration and exploitation in AFSSA

In swarm intelligence algorithms, maintaining a balance between exploration and exploitation is critical to prevent premature convergence and to effectively locate global optima. The proposed AFSSA algorithm explicitly addresses this trade-off through its fuzzy dynamic control mechanism combined with multi-modal sampling distributions.

Exploration in AFSSA is primarily realized by sampling candidate positions from different probability distributions centered around the global best particle. Particularly, the Trapezoidal distribution produces a broad spread of particles, enabling a global search around promising regions. The Gaussian distribution, with a moderate spread, balances exploration and local refinement, while the Triangular distribution restricts particles to a narrow vicinity of the best solution, favoring exploitation.

The Mamdani fuzzy inference system plays a pivotal role by dynamically adjusting the acceleration coefficients c_1 and c_2 based on the Curve Variation Density Ratio, which quantifies local landscape smoothness and complexity. High values of this ratio prompt the algorithm to increase exploration tendencies, whereas low values focus the search on local exploitation by strengthening the influence of personal and global best positions.

This adaptive mechanism allows AFSSA to smoothly transition between exploration and exploitation phases in response to evolving problem characteristics, thus improving convergence reliability and avoiding stagnation in local optima. Empirical results demonstrate that this balance significantly enhances optimization performance on high-dimensional and complex functions.

4 Numerical experiment and discussion

In this section, we present a comprehensive evaluation of the proposed Adaptive Fuzzy Swarm-based Search Algorithm (AFSSA) when integrated with Particle Swarm Optimization (PSO) and Golden Search Optimization (GSO). The performance analysis is carried out using four categories of benchmark functions: (1) unimodal functions (F1–F7) to test exploitation and convergence precision, (2) multimodal functions (F8–F13) to evaluate exploration and diversity maintenance, (3) hybrid composition functions (F14–F23) to assess adaptive balancing of search strategies, and (4) the CEC 2019 test suite (C1–C10) to examine robustness in complex, real-world-like landscapes [1, 23, 28].

We first describe the parameter settings and experimental setup to ensure fair comparisons among all algorithms. Then, qualitative assessments are provided through convergence plots, fitness trajectories, and search behavior visualizations to illustrate the dynamics of AFSSA-enhanced algorithms. This is followed by a quantitative performance evaluation against the original PSO and GSO algorithms as well as other state-of-the-art metaheuristics, supported by statistical analyses to identify significant differences in performance. Finally, a practical example of data clustering is presented to demonstrate the applicability of AFSSA in real-world optimization problems, highlighting its ability to find compact and well-separated clusters in complex, high-dimensional datasets. Together, these evaluations provide a detailed and rigorous examination of the effectiveness, reliability, and practical applicability of AFSSA in complex optimization tasks across diverse problem characteristics.

4.1 Parameter settings

All implementations of the optimization algorithms in this study were carried out using R version 4.4.1. The R environment also utilized the "smoof" package to access benchmark functions for optimization problems and statistical analysis. The experiments were executed on a system equipped with an Intel(R) Core(TM) i5-6400 CPU at 2.70 GHz, 16 GB of RAM, and a 64-bit Windows 10 Pro operating system.

To ensure an equitable comparison, the number of iterations and the number of samples (particles) were consistently set to 1000 and 80, respectively, for all optimization methods. Notably, the proposed AFSSA approach is parameter-free, and only these two settings are required. The AFSSA technique was integrated with the PSO [5] and GSO [25] algorithms and evaluated against their original versions. Additionally, for a more comprehensive assessment, the AFSSA performance was compared with other state-of-the-art algorithms, including TSA [18], GWO [22], and SCA [21]. All other algorithm-specific parameters were selected according to the configurations outlined in the original studies, as summarized in Table 3.

4.2 Qualitative evaluation of AFSSA-enhanced PSO and GSO algorithms

In this section, qualitative evaluations of the performance of the PSO and GSO algorithms enhanced with AFSSA are presented. The evaluations are conducted using four different graphs—Convergence, Average Fitness, Search History, and the Path of the First Search Agent—based on two functions: $F1$ (unimodal) and $F9$ (multimodal), as shown in Figures 2 and 3, respectively. The results clearly demonstrate that the combination of AFSSA with PSO and GSO significantly outperforms the original versions of these algorithms for both functions.

Convergence Graphs: For the F1 function, the AFSSA-PSO and AFSSA-GSO combinations converge rapidly to the optimal solution, consistently reducing the cost. In F9, these algorithms are also capable of avoiding local optima and progressing towards the global optimum, whereas the original PSO and GSO versions face difficulties in more complex search spaces.

Table 3: Parameter settings of optimization algorithms

Algorithm	**Parameter**	**Value**
GSO [25]	C_1, C_2	Random values in [0, 2]
PSO [5]	$C_1 = C_2$	2
	Inertia weight w	[0.4, 0.9]
TSA [18]	Parameter $P_{\min}$	1
	Parameter $P_{\max}$	4
GWO [22]	Convergence constant a	[2, 0]
SSA [21]	C_1, C_2	Random values in [0, 1]

Average Fitness Graphs: In both F1 and F9, the AFSSA-PSO and AFSSA-GSO combinations show a consistent decrease in average fitness, approaching the optimal point more effectively. This faster and more consistent reduction indicates better performance in terms of solution quality compared to the original versions.

Search History Graphs: In both functions F1 and F9, the AFSSA combinations cover the search space more effectively and can identify optimal points. AFSSA helps to escape from local optima and conduct a broader search, whereas the original versions remain more concentrated in the regions close to the optimal point.

Trajectory of 1st Search Agent: For both F1 and F9, the trajectory of the first search agent in AFSSA-PSO and AFSSA-GSO shows faster and more effective movement towards the optimal points. In contrast, the original PSO and GSO exhibit slower and less efficient trajectories.

4.3 Performance analysis of AFSSA-PSO and AFSSA-GSO on benchmark and CEC-2017 functions

In this section, the performance of the AFSSA-PSO and AFSSA-GSO algorithms is evaluated in comparison with the original versions of PSO and GSO, as well as other algorithms such as the Tunicate Swarm Algorithm (TSA) [18], Grey Wolf Optimizer (GWO) [22], and Salp Swarm Algorithm (SSA) [21]. A total of 33 benchmark functions were employed for evaluation, each of which is discussed in detail in the corresponding sections.

First, a scalability analysis was conducted using functions F_1 to F_{13}, which are scalable and allow for dimensional adjustments. These functions were tested in four separate experiments with dimensions of 30, 100, and 500. The results are presented in Tables 4 to 6. This experiment aimed to determine whether integrating AFSSA with swarm intelligence algorithms can maintain the algorithm's search capability when addressing higher-dimensional problems.

Focusing on the results shown in Table 4, the AFSSA-PSO and AFSSA-GSO algorithms demonstrate superior performance compared to other optimizers. For instance, in function F_1 with 30 dimensions, AFSSA-GSO achieves an average value of $5.1422E-188$, significantly outperforming GSO ($5.4203E-102$) and PSO ($4.3260E+02$). Similarly, AFSSA-PSO shows competitive results, achieving $4.3907E-172$. This trend continues across other functions, with AFSSA-GSO and AFSSA-PSO consistently ranking among the top performers.

Subsequently, Table 5 presents the results of AFSSA-PSO, AFSSA-GSO, and the other compared algorithms for functions F_1 to F_{13} in 100 dimensions. The results indicate that AFSSA-PSO and AFSSA-GSO significantly outperform the other algorithms in most of the functions. For example, in function F_2, AFSSA-GSO achieves an average value of $3.0517E-109$, while GSO and PSO achieve $2.4897E-42$ and $4.0450E+02$, respectively. This demonstrates the robustness of AFSSA-GSO and AFSSA-PSO in handling higher-dimensional problems.

Table 6 shows the results for dimension 500, where the AFSSA-PSO and AFSSA-GSO algorithms continue to provide strong results for unimodal functions. They maintain their search capability in the resonance components and outperform other algorithms. For instance, in function F_1, AFSSA-GSO achieves an average value of $6.5792E-191$, while GSO and PSO achieve $5.2401E-60$ and $1.1141E+05$, respectively. This highlights the ability of AFSSA-GSO and AFSSA-PSO to maintain high performance even as the problem dimensionality increases.

Table 7 shows the results for functions F_{14} to F_{23}, which have fixed dimensions. The results in this table show the excellent performance of the AFSSA-PSO and AFSSA-GSO algorithms in solving problems F_{14} to F_{23}, achieving near-optimal solutions in most cases and ranking second in only 2 out of 10 cases. For instance, in function F_{14}, both

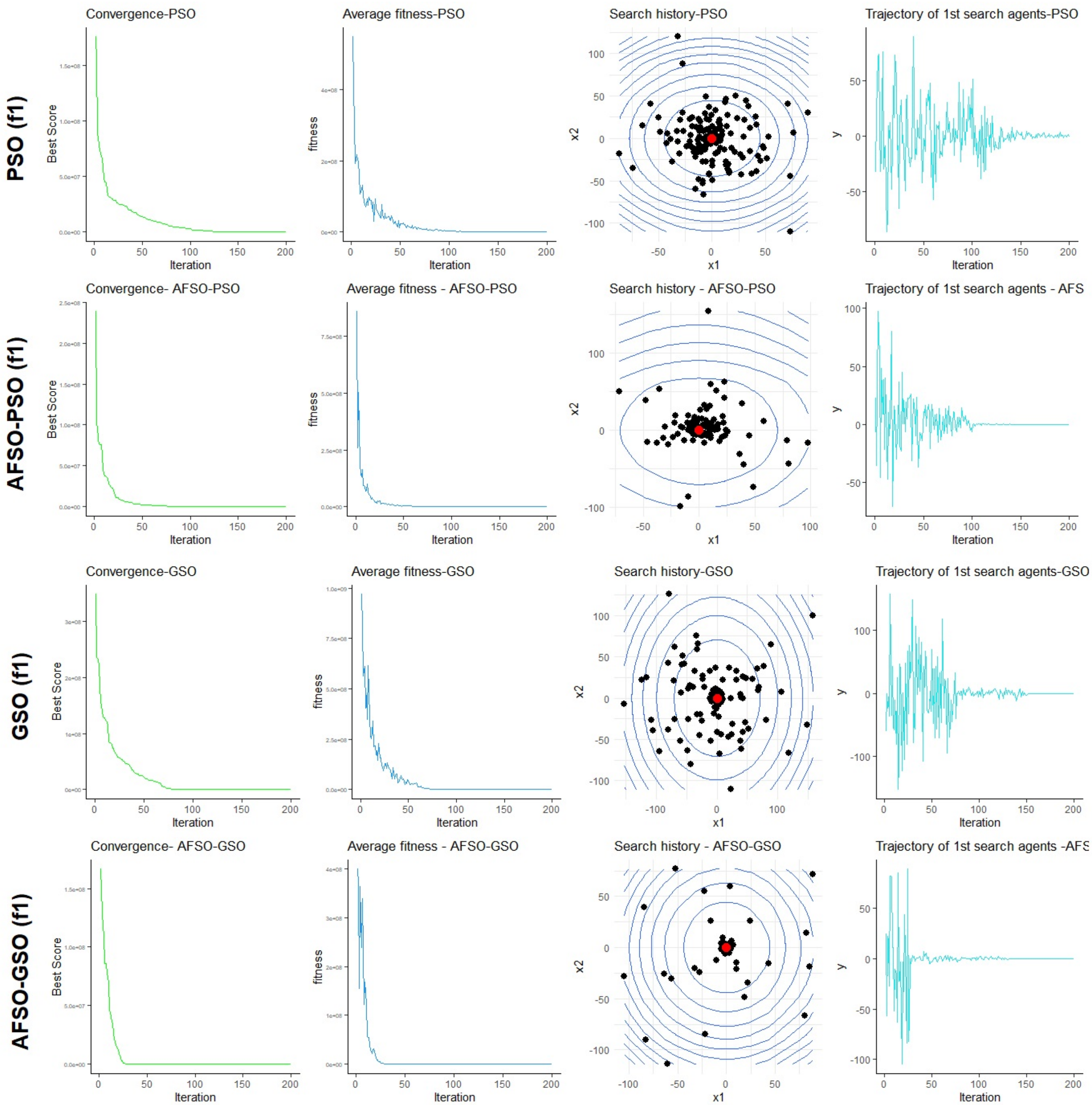

Figure 2: Performance comparison between PSO and AFSSA-PSO, as well as GSO and AFSSA-GSO on F1

AFSSA-GSO and AFSSA-PSO achieve the optimal value of $9.9800E-01$, matching the performance of other algorithms but with significantly lower standard deviations, indicating greater consistency.

For further analysis, the CEC2019 benchmark functions were utilized due to their high complexity and difficulty in solving. Table 8 presents the evaluation results obtained using the CEC2019 criteria. The table underscores the remarkable performance of the AFSO-PSO and AFSO-GSO algorithms compared to other optimizers, delivering superior results in 8 out of 10 functions. For example, in function $C1$, AFSSA-GSO achieves an average value of $3.5453E+04$, significantly outperforming GSO ($6.6207E+05$) and PSO ($4.2513E+05$). Achieving such outcomes in these functions

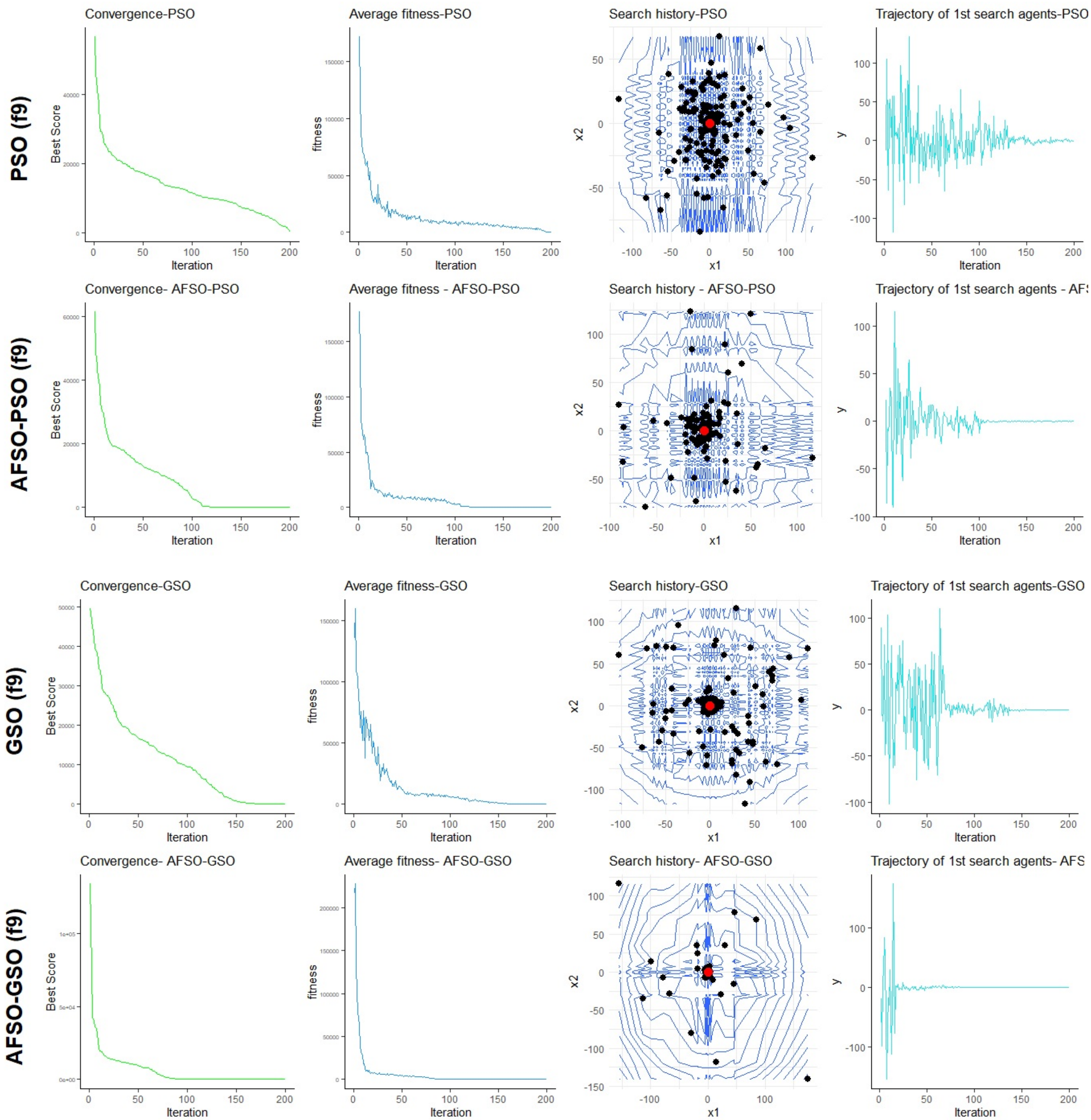

Figure 3: Performance comparison between PSO and AFSSA-PSO, as well as GSO and AFSSA-GSO on F9

necessitates a balanced performance in both the exploration and exploitation phases. The AFSO-PSO and AFSO-GSO algorithms demonstrate exceptional capability in maintaining this balance and automatically adapting between these two phases.

In conclusion, the results presented in Tables 4 to 8 demonstrate the superior performance of the AFSSA-PSO and AFSSA-GSO algorithms across a wide range of benchmark functions and problem dimensions. Their ability to maintain high performance in both low and high-dimensional problems, as well as their balanced exploration and exploitation capabilities, make them highly effective optimization tools.

Table 4: Comparison of results on F1–F13 with 30 dimensions

Function	Metric	AFSSA-GSO	GSO	AFSSA-PSO	PSO	TSA	GWO	SSA
F1	AVG	**5.1422E-188**	5.4203E-102	4.3907E-172	4.3260E+02	1.1885E-52	4.7808E-70	9.1604E-11
	STD	**1.9683E-191**	7.6003E-103	2.5636E-172	5.8347E+01	3.6265E-53	1.2008E-71	1.4182E-11
F2	AVG	**7.6301E-133**	1.4336E-57	1.1503E-108	1.3555E+04	1.1705E-30	3.9508E-41	4.7903E-02
	STD	**4.2373E-134**	1.7205E-57	4.7407E-108	4.3651E+03	4.7605E-31	3.9208E-42	6.7507E-02
F3	AVG	**3.6601E-165**	4.9708E-27	4.5764E-69	2.6703E+03	7.3880E-15	5.9907E-20	7.0508E+01
	STD	**1.5033E-167**	2.0307E-28	4.4377E-70	8.3371E+02	4.7005E-15	1.9708E-19	2.6349E+01
F4	AVG	**2.4201E-76**	2.6165E-25	2.6572E+01	3.5474E+00	6.1374E-04	1.9407E-17	4.0451E+00
	STD	**5.4271E-76**	6.7105E-25	8.8618+00	9.1452E-01	2.9205E-04	3.0007E-18	2.6534E-01
F5	AVG	4.2523E+01	4.1754E+02	1.3318E+02	4.2587E+03	**1.6023E+01**	8.3858E+02	2.4426E+02
	STD	1.3627E+01	3.5297E+01	3.6594E+01	1.2813E+02	**3.6682E+01**	2.6288E+01	1.9681E+02
F6	AVG	**1.7801E-05**	1.8991E-01	4.2003E-03	1.2304E-01	3.6023E-03	4.1003E-01	9.6701E-02
	STD	**9.1003E-05**	6.1304E-02	2.2701E-03	3.1202E-03	7.0603E-03	3.0604E-02	2.4201E-02
F7	AVG	**3.1203E-04**	4.2504E-05	2.6101E-03	5.3602E-02	3.5103E-03	4.256E-04	5.7004E-02
	STD	**4.5804E-05**	2.2203E-05	3.0601E-03	5.1302E-03	1.3504E-03	2.1601E-04	2.5203E-02
F8	AVG	-2.5272E+04	-9.5843E+04	-1.1901E+04	-9.0702E+04	**-6.3804E+03**	-6.0401E+04	-7.6203E+04
	STD	9.0003E+02	7.2604E+03	1.2702E+02	5.4601E+03	**6.8403E+02**	8.0804E+03	8.7401E+03
F9	AVG	**0.00E+00**	0.00E+00	0.00E+00	4.3001E+02	5.3202E+02	2.7203E-01	3.1104E+01
	STD	**0.00E+00**	0.00E+00	0.00E+00	3.9802E+01	1.2503E+01	3.3504E+00	2.4601E+01
F10	AVG	**7.9001E-16**	3.3302E-11	5.2103E-12	1.7802E-01	3.6043E-08	7.6202E-11	3.8501E-02
	STD	**0.0000E+00**	4.6662E-11	1.8203E-12	5.7902E-01	3.3302E-09	4.8803E-10	8.1002E-01
F11	AVG	**0.0000E+00**	0.0000E+00	7.4002E-03	1.7302E+02	6.5903E-03	1.4202E-03	1.1202E-02
	STD	**0.0000E+00**	0.0000E+00	1.9602E-02	3.2102E+01	7.0703E-03	4.6002E-03	1.3302E-02
F12	AVG	**3.1702E-09**	3.4802E-03	1.2202E-03	1.5302E+01	8.0203E-08	2.5402E-02	3.9402E+00
	STD	**8.1502E-09**	1.9102E-02	1.5802E-03	9.9202E+01	4.2003E+00	1.7602E-02	2.0102E-06
F13	AVG	**1.9402E-02**	3.7202E-01	4.5502E-01	5.7502E+02	2.6102E-01	2.8202E-01	7.5302E-01
	STD	**4.4102E-01**	7.5102E-01	4.3102E-01	3.8202E+01	2.6202E-01	2.3102E-01	2.4202E-02

4.4 Statistical test

To identify significant differences between the outcomes of metaheuristic algorithms, statistical tests can be employed for a comprehensive comparison. In this study, we have applied the Paired T-test for statistical analysis, and the resulting P-values are presented in Tables 13 to 18. A P-value greater than 0.05 suggests that no substantial differences exist between the results of the two optimizers under comparison based on this statistical method, whereas a P-value smaller than 0.05 indicates the presence of statistically meaningful differences in their performances. Tables 9, 10, and 11 display the P-values for functions 1 through 13 with dimensions of 30, 100, and 500, while benchmarks 14 to 23 are summarized in Table 12. Lastly, the P-values for the CEC2019 functions are provided in Table 13.

Based on the P-values presented in Tables 9, 10, and 11, which pertain to the scalability tests, significant differences were observed in most cases. AFSSA-GSO and AFSSA-PSO demonstrated statistically superior performance compared to both their original versions and other optimization algorithms. However, in benchmarks 9 and 11, similar results were found between AFSSA-GSO, AFSSA-PSO, and GSO, where the performance was nearly identical across all dimensions. Overall, for benchmarks F1 to F13 across all dimensions, AFSSA-GSO consistently outperformed the other algorithms, delivering strong and statistically significant results.

Table 12 presents the P-values obtained from criteria 14 to 23. According to this statistical analysis, AFSSA-GSO and AFSSA-PSO again demonstrated significantly better performance, producing solutions that were notably superior to those of the other compared algorithms. This further emphasizes their enhanced search capabilities.

Finally, Table 13 presents the T-test results using the CEC2019 functions. A detailed analysis of the table reveals that AFSSA-GSO and AFSSA-PSO exhibit statistically significant differences in performance, with their superior results becoming more pronounced as the problem complexity increases. This suggests that as the problem difficulty grows, the robust performance of AFSSA-GSO and AFSSA-PSO becomes increasingly crucial, allowing them to explore and exploit the problem space more efficiently. Overall, this statistical comparison demonstrates that combining the AFSSA

Table 5: Comparison of results on F1–F13 with 100 dimensions

Function	Metric	AFSSA-GSO	GSO	AFSSA-PSO	PSO	TSA	GWO	SSA
F1	AVG	**5.9557E-181**	1.6923E-72	1.3604E-168	5.4657E+02	3.4521E-57	2.0141E-34	9.3287E-03
	STD	**4.2067E-183**	4.1595E-71	9.7370E-171	2.5047E+03	2.6961E-49	5.3530E-34	1.1296E-02
F2	AVG	**3.0517E-109**	2.4897E-42	6.3166E-103	4.0450E+02	7.5314E-18	7.0736E-21	1.4202E+01
	STD	**1.6632E-107**	2.6546E-40	3.4717E-102	2.4642E+01	1.1250E-17	3.2441E-21	4.4149E-07
F3	AVG	**2.8413E-145**	2.1714E-106	7.2630E-105	4.3730E+04	4.7772E+02	1.6589E-01	2.2013E+04
	STD	**1.5736E-144**	5.6621E-106	1.2114E-105	1.9503E+04	1.0744E+03	3.5655E-01	1.0357E+04
F4	AVG	**1.3622E-97**	5.5043E-11	7.1034E-21	3.2857E+07	2.5133E+01	2.3447E-04	6.1967E+01
	STD	**4.6441E-95**	1.7207E-10	3.02.1E-20	3.6354E+06	7.1173E+00	1.5724E-04	4.4983E+00
F5	AVG	**3.7486E+00**	6.7271E+01	5.2259E+01	6.1703E+04	5.7649E+01	8.1381E+01	4.0742E+02
	STD	**1.2445E+00**	6.9351E+01	6.7453E+01	4.2225E+04	3.6320E+01	7.1719E+01	8.6357E+02
F6	AVG	**2.2249E-02**	1.3124E-01	4.7387E-01	5.2443E+03	3.8267E+01	9.6041E+00	5.8338E+00
	STD	**4.4214E-01**	6.9125E-01	4.67342E-01	4.3328E+03	2.2662E-01	3.6285E+00	3.2279E+00
F7	AVG	**4.7322E-05**	7.5076E-04	8.9442E-04	3.8539E+00	1.5241E-02	1.3551E-03	7.9271E-01
	STD	**6.4222E-05**	3.8659E-04	1.5017E-03	7.9129E+00	4.5277E-03	6.1291E-04	1.6663E-01
F8	AVG	**-4.1240E+04**	-2.7128E+04	-3.9752E+04	-2.478E+03	-1.3258E+04	-1.5027E+04	-2.5143E+04
	STD	**1.3124E+02**	1.6274E+03	4.9440E+03	1.6757E+03	8.2330E+03	2.7554E+03	1.6126E+03
F9	AVG	**0.0000E+00**	0.0000E+00	0.0000E+00	3.2478E+02	8.1374E-06	1.6442E-01	1.4030E-02
	STD	**0.0000E+00**	0.0000E+00	0.0000E+00	5.4512E+01	1.5302E-05	8.7745E-01	3.1595E-01
F10	AVG	**7.2032E-16**	7.7458E-11	4.7464E-11	4.8211E+04	6.1401E-12	7.2342E-09	5.3412E-04
	STD	**0.0000E+00**	1.3532E-10	2.2889E-10	3.2782E+03	3.4662E-11	4.9242E-07	1.3041E-04
F11	AVG	**0.0000E+00**	0.0000E+00	0.0000E+00	7.4748E+03	2.5595E-03	7.87E-04	1.0333E-01
	STD	**0.0000E+00**	0.0000E+00	0.0000E+00	2.4623E+02	6.7558E-03	3.2205E-03	3.0271E-02
F12	AVG	**3.4261E-03**	1.9518E-01	6.3245E-03	1.1444E+03	9.6302E-01	1.8359E-01	1.0272E+00
	STD	**5.1432E-03**	5.7294E-01	3.3293E-03	4.5425E+02	3.0591E-01	6.5361E-01	2.7321E-01
F13	AVG	**1.5031E-03**	2.3441E+00	5.3874E-01	4.6222E+02	1.4435E-01	5.8020E+00	1.6218E-02
	STD	**1.8339E-02**	1.2425E+00	3.4141E-01	4.3756E+02	1.9278E+00	3.9247E-01	1.0468E-01

method with crowd intelligence algorithms enhances the ability and efficiency to solve various problems, particularly as the problem complexity escalates.

Table 9: P-value results from the T-test for F1–F13 benchmark with 30D

AGSO-GSO					
F	**GSO**	**PSO**	**TSA**	**GWO**	**SSA**
F1	3.25E-01	3.14E-05	3.14E-02	3.13E-04	3.17E-04
F2	3.12E-03	2.13E-05	2.23E-03	4.57E-03	3.35E-05
F3	3.23E-02	2.46E-05	3.57E-03	2.46E-05	4.35E-04
F4	3.35E-02	4.15E-05	4.15E-05	4.36E-05	1.12E-03
F5	8.90E-02	1.23E-04	2.35E-04	2.35E-05	3.12E-03
F6	3.21E-03	2.12E-03	3.14E-03	3.14E-05	1.57E-04
F7	3.12E-02	1.68E-03	3.14E-04	2.35E-05	3.57E-03
F8	4.11E-03	7.12E-03	3.14E-04	2.46E-05	3.57E-03
F9	1.00E+00	2.35E-05	2.68E-02	1.23E+01	4.57E-04
F10	3.46E-03	3.15E-05	3.15E-05	3.16E-05	2.68E-04
F11	1.00E+00	2.35E-05	2.35E-04	2.46E-05	9.12E-04
F12	1.02E-01	2.35E-05	2.35E-04	4.57E-03	3.57E-03
F13	1.68E-01	3.14E-05	3.15E-05	3.14E-05	3.14E-03

AGSO-PSO					
F	**GSO**	**PSO**	**TSA**	**GWO**	**SSA**
F1	7.61E-02	3.59E-05	1.46E-02	3.85E-03	2.57E-02
F2	2.12E-03	2.33E-05	3.23E-04	5.57E-03	2.25E-05
F3	4.23E-03	1.46E-05	5.37E-03	2.14E-03	2.14E-03
F4	4.11E-03	1.35E-05	1.35E-03	3.26E-03	1.32E-02
F5	7.92E-02	8.45E-03	4.15E-03	5.14E-04	1.92E-03
F6	2.65E-02	9.56E-03	4.56E-03	1.26E-03	1.47E-02
F7	3.15E-03	3.15E-03	3.14E-03	2.16E-03	2.47E-03
F8	3.14E-02	3.11E-03	3.48E-03	3.14E-03	2.23E-03
F9	1.00E+00	3.23E-05	3.68E-02	1.09E-01	5.47E-01
F10	2.45E-03	3.76E-04	1.76E-03	2.65E-02	2.95E-02
F11	4.85E-03	3.35E-05	4.45E-03	2.46E-02	8.42E-02
F12	1.02E-01	2.76E-04	2.45E-03	5.58E-01	3.47E-03
F13	1.61E-01	2.35E-05	3.15E-03	2.35E-03	2.67E-03

Table 6: Comparison of results on F1–F13 with 500 dimensions

Function	Metric	AFSSA-GSO	GSO	AFSSA-PSO	PSO	TSA	GWO	SSA
F1	AVG	**6.5792E-191**	5.2401E-60	2.9657E-160	1.1141E+05	4.2552E-11	2.5469E-15	3.1815E-04
	STD	**0.0000E+00**	6.2244E-60	0.0000E+00	2.1372E+04	5.1670E-11	1.3208E-15	2.3371E-03
F2	AVG	4.8286E-97	5.7163E-36	**1.0358E-105**	9.0787E+02	9.7263E-09	5.0679E-09	3.4354E-02
	STD	2.6855E-96	3.7044E-36	**5.8289E-105**	1.3624E+02	8.3703E-09	1.8528E-09	1.4325E-01
F3	AVG	**1.1522E-206**	4.5723E-103	2.7522E-141	2.1530E+06	9.2447E-05	8.5456E-04	5.9222E-05
	STD	**5.4177E-203**	1.6802E-104	6.2154E-142	3.0433E+05	1.3021E-05	3.4447E-04	2.8685E-05
F4	AVG	**7.1954E-95**	8.4542E-32	7.4227E-78	7.6402E+01	9.8325E-01	5.2445E-01	3.3664E-01
	STD	**4.1452E-94**	2.0121E-30	2.5021E-77	9.0412E+00	2.3769E-01	6.3215E+00	2.4228E-01
F5	AVG	1.6130E+02	4.4189E+02	4.385E+02	1.8977E+08	**5.3325E+01**	5.0663E+02	5.6042E+02
	STD	1.9870E+01	7.8033E+01	2.4744E+01	1.1558E+07	**2.0315E+01**	4.4292E+01	9.5836E+01
F6	AVG	**1.0189E-01**	5.8328E+01	9.6221E+00	1.2042E+05	9.7440E+01	8.7355E+01	3.4270E+04
	STD	**1.2142E-01**	2.6211E+00	1.3078E+00	2.5435E+04	1.8526E+00	2.1220E+00	2.3312E+03
F7	AVG	**3.2291E-05**	1.6172E-03	1.2524E-03	1.8337E+03	3.1125E-01	7.8669E-03	5.4285E+01
	STD	**2.9275E-05**	5.3178E-04	1.2559E-03	7.2322E+02	1.0424E-01	2.5046E-03	7.3250E+00
F8	AVG	-2.1501E+05	-1.1212E+05	-2.0541E+05	-7.8086E+04	**-2.5040E+05**	-6.2532E+04	-8.6720E+04
	STD	1.4320E+02	6.8245E+03	1.7302E+04	3.2110E+03	**2.547E+03**	9.7300E+03	5.3755E+03
F9	AVG	**0.0000E+00**	0.0000E+00	0.0000E+00	3.2058E+03	5.8041E+03	2.1627E+00	1.9548E+03
	STD	**0.0000E+00**	0.0000E+00	0.0000E+00	1.7515E+02	4.9440E+02	3.2320E+00	1.2020E+02
F10	AVG	**8.1297E-16**	8.5405E-15	4.1272E-15	1.7221E+01	1.6308E-07	6.8344E-09	1.1258E-01
	STD	**0.000E+00**	2.2415E-15	2.3235E-15	7.7458E-01	5.4300E-08	1.1440E-09	3.2050E-01
F11	AVG	**0.0000E+00**	0.0000E+00	0.0000E+00	1.2352E+03	6.4105E-03	1.8447E-03	2.9305E+02
	STD	**0.0000E+00**	0.0000E+00	0.0000E+00	2.4410E+02	1.5155E-02	7.0333E-03	2.2023E+01
F12	AVG	**2.1285E-03**	2.7424E-01	1.3628E-02	1.2140E+08	1.2013E+01	7.1630E-01	1.0118E+02
	STD	**4.3001E-03**	1.4494E-02	3.2087E-03	1.6435E+08	1.2950E+01	3.1972E-02	1.6580E+01
F13	AVG	**4.4242E-02**	4.3895E+01	4.1880E+00	5.9245E+08	1.3180E+03	4.60E+01	1.2005E+06
	STD	**7.2110E-02**	3.6320E-01	1.5482E+00	4.5262E+08	8.1790E+02	5.2547E+00	4.0345E+05

Table 10: P-value results from the T-test for F1-F13 benchmark with 100D

AGSO-GSO					
F	**GSO**	**PSO**	**TSA**	**GWO**	**SSA**
F1	6.25E-03	3.78E-05	1.36E-04	2.65E-04	4.22E-03
F2	4.18E-03	2.06E-05	2.02E-04	3.12E-04	3.45E-04
F3	9.82E-02	4.32E-05	1.32E-03	3.76E-05	6.45E-04
F4	2.56E-02	2.95E-05	1.35E-04	2.45E-05	8.12E-04
F5	1.24E-02	1.89E-04	2.92E-04	3.53E-05	1.19E-03
F6	3.54E-03	1.22E-04	1.91E-03	2.36E-05	1.22E-04
F7	2.57E-02	1.64E-04	1.55E-04	2.67E-05	3.58E-03
F8	4.78E-01	6.92E-04	2.02E-02	2.76E-03	3.15E-02
F9	1.00E+00	2.91E-05	2.56E-02	1.11E+01	3.65E-04
F10	3.92E-03	2.72E-05	2.53E-04	2.11E-05	2.48E-04
F11	1.00E+00	3.11E-05	2.79E-04	2.32E-05	9.11E-04
F12	8.42E-02	2.13E-05	1.89E-04	3.46E-03	3.73E-03
F13	1.72E-01	2.13E-05	2.13E-05	2.13E-05	2.14E-03

AGSO-PSO					
F	**GSO**	**PSO**	**TSA**	**GWO**	**SSA**
F1	7.36E-02	3.93E-05	1.45E-02	4.02E-03	2.36E-02
F2	2.47E-03	2.51E-05	3.23E-04	5.42E-03	2.19E-05
F3	4.02E-03	1.55E-05	5.36E-03	1.58E-03	3.15E-03
F4	1.98E-03	2.13E-05	2.14E-03	2.76E-03	1.45E-02
F5	6.84E-02	4.52E-03	4.03E-03	4.91E-04	1.76E-03
F6	2.49E-02	2.14E-02	4.48E-03	1.29E-03	2.12E-02
F7	2.42E-03	1.59E-03	2.87E-03	1.46E-03	2.47E-03
F8	2.64E-01	8.32E-02	2.13E-02	3.51E-02	2.37E-02
F9	1.00E+00	3.67E-05	3.92E-02	1.11E+01	5.21E-01
F10	2.35E-03	3.47E-04	1.85E-03	2.64E-02	3.06E-02
F11	1.00E+00	3.56E-05	4.35E-03	2.58E-02	9.05E-02
F12	1.09E-01	2.88E-04	2.69E-03	5.54E-01	3.69E-03
F13	1.63E-01	2.48E-05	3.28E-03	2.48E-03	2.72E-03

Table 7: Comparison of results on F14–F23

Function	Metric	AFSSA-GSO	GSO	AFSSA-PSO	PSO	TSA	GWO	SSA
F14	AVG	**9.9801E-01**	9.9817E-01	9.9815E-01	9.9843E-01	9.9804E-01	9.9812E-01	9.9800E-01
	STD	**7.3642E-14**	4.8143E-08	7.4612E-13	3.4456E-11	4.5923E-16	3.4487E-17	3.2964E-09
F15	AVG	**3.5201E-05**	6.7012E-02	6.5512E-04	5.3701E-02	3.2500E-02	3.5412E-03	6.8700E-03
	STD	**2.8765E-04**	5.1354E-03	2.1531E-03	3.3321E-04	5.9723E-03	6.3102E-02	4.7031E-03
F16	AVG	**-1.0300E+00**	-1.0300E+00	-1.0300E+00	-1.0300E+00	-1.0300E+00	-1.0300E+00	-1.0300E+00
	STD	**8.2464E-15**	4.8745E-11	2.5512E-09	6.3254E-14	8.4932E-03	8.3421E-08	7.2512E-14
F17	AVG	**3.9801E-01**	3.9801E-01	3.9801E-01	3.9801E-01	3.9801E-01	3.9801E-01	3.9801E-01
	STD	**2.7432E-08**	7.1154E-03	1.2132E-04	3.3012E-05	5.6731E-05	2.3801E-02	4.7713E-06
F18	AVG	**3.0000E+00**	3.0000E+00	3.0000E+00	3.0000E+00	9.3000E+00	3.0000E+00	3.0000E+00
	STD	**2.7123E-13**	3.5924E-11	2.6534E-12	7.8462E-15	2.3321E-15	5.8631E-13	3.9021E-15
F19	AVG	**-3.8600E+00**	-3.8600E+00	-3.8600E+00	-3.8600E+00	-3.8600E+00	-3.8600E+00	-3.8600E+00
	STD	**7.2056E-15**	4.3212E-12	1.2854E-15	4.8800E-14	3.5562E-01	3.0621E-12	1.8112E-12
F20	AVG	**-3.2800E+00**	-3.1600E+00	-3.2200E+00	-3.2600E+00	-3.2100E+00	-3.2500E+00	-3.2300E+00
	STD	**4.2132E-02**	3.2145E-01	1.8912E-02	2.4432E-01	2.1345E-02	7.3641E-01	2.7734E-02
F21	AVG	-9.8100E+00	-9.7600E+00	-9.8100E+00	-2.1300E+00	**-1.0100E+01**	-8.6300E+00	-9.0500E+00
	STD	9.1345E-01	2.7045E+00	1.4400E+00	3.9500E+00	**3.3201E-05**	1.3012E+00	2.6611E+00
F22	AVG	-4.2395E+00	-9.2924E+00	-9.4375E+00	-1.3628E+00	**-1.0000E+01**	-1.3200E+00	-9.2163E+00
	STD	1.9757E+00	4.2699E+00	5.8424E+00	2.9115E+00	**1.4873E-06**	2.6366E-04	3.2321E+00
F23	AVG	-8.2635E+00	-1.0841E+01	-9.8163E+00	-1.7411E+00	**-1.0300E+00**	-6.1163E+00	-8.3241E+00
	STD	1.6329E-00	1.2663E-01	1.9711E+00	1.6445E+00	**2.3258E-04**	1.2339E-04	2.4100E+00

Table 11: P-value results from the T-test for F1-F13 benchmark with 500D

AGSO-GSO					
F	GSO	PSO	TSA	GWO	SSA
F1	6.25E-02	6.14E-05	6.25E-04	2.53E-04	6.25E-03
F2	5.10E-03	5.11E-05	5.10E-04	5.22E-04	5.12E-04
F3	1.12E-01	4.21E-05	1.21E-03	3.44E-05	7.52E-04
F4	2.74E-02	3.05E-05	1.46E-04	2.56E-05	8.91E-04
F5	1.18E-02	1.92E-04	3.11E-04	3.21E-05	1.23E-03
F6	3.64E-03	1.43E-04	2.01E-03	2.48E-05	1.35E-04
F7	2.47E-02	1.82E-04	1.41E-04	2.73E-05	3.71E-03
F8	3.51E-01	3.51E-02	3.44E-02	2.68E-03	3.32E-02
F9	1.00E+00	3.11E-05	2.63E-02	1.10E+01	3.89E-04
F10	3.83E-03	3.41E-05	6.46E-04	2.25E-05	2.58E-04
F11	1.00E+00	3.22E-05	2.67E-04	2.51E-05	9.34E-04
F12	8.11E-02	2.53E-05	2.03E-04	3.56E-03	3.91E-03
F13	1.85E-01	2.67E-05	4.17E-05	4.17E-05	4.72E-03

AGSO-PSO					
F	GSO	PSO	TSA	GWO	SSA
F1	7.56E-02	4.08E-05	1.53E-02	4.24E-03	2.57E-02
F2	2.56E-03	2.67E-05	3.01E-04	5.11E-03	2.33E-05
F3	3.87E-03	1.71E-05	5.47E-03	1.61E-03	3.32E-03
F4	1.87E-03	1.32E-05	1.51E-03	2.83E-03	1.56E-02
F5	7.02E-02	8.21E-03	4.28E-03	5.12E-04	1.94E-03
F6	2.11E-02	2.10E-02	3.72E-03	2.13E-03	2.75E-02
F7	2.32E-03	2.28E-03	3.15E-03	1.63E-03	2.56E-03
F8	4.80E-02	4.23E-02	4.56E-02	4.67E-02	2.51E-02
F9	1.00E+00	3.58E-05	4.01E-02	1.10E+01	5.38E-01
F10	2.45E-03	3.63E-04	1.98E-03	2.72E-02	3.14E-02
F11	1.00E+00	3.87E-05	4.50E-03	2.69E-02	9.34E-02
F12	1.16E-01	2.92E-04	2.79E-03	5.71E-01	3.82E-03
F13	3.79E-01	3.56E-05	3.43E-03	3.37E-03	3.86E-03

Table 12: The P-value results from the T-test for F14-F23 benchmarks

AGSO-GSO					
F	GSO	PSO	TSA	GWO	SSA
F14	4.17E-04	3.89E-05	1.09E-04	2.61E-04	5.19E-03
F15	2.89E-04	2.71E-05	2.48E-04	2.62E-04	2.62E-04
F16	3.54E-04	3.53E-05	1.12E-04	2.53E-05	3.28E-04
F17	2.39E-04	2.96E-05	1.10E-04	2.81E-05	9.14E-04
F18	1.30E-04	1.61E-04	3.18E-04	3.06E-05	1.00E-03
F19	4.22E-04	4.21E-04	4.21E-04	4.26E-04	4.21E-04
F20	2.29E-04	1.46E-04	1.30E-04	2.59E-05	3.91E-03
F21	4.43E-04	6.03E-04	1.95E-04	2.18E-03	2.92E-02
F22	7.16E-04	3.23E-05	2.66E-04	3.76E+04	3.95E-04
F23	3.90E-04	2.80E-05	2.62E-04	2.43E-05	2.37E-04

AGSO-PSO					
F	GSO	PSO	TSA	GWO	SSA
F14	3.52E-04	1.59E-05	4.43E-04	2.32E-04	4.87E-04
F15	2.82E-03	2.91E-05	3.46E-04	5.61E-03	2.53E-05
F16	3.18E-03	1.47E-05	5.08E-03	1.11E-03	3.22E-03
F17	2.21E-03	1.14E-03	1.70E-03	2.44E-03	1.37E-03
F18	6.88E-02	7.72E-03	4.05E-03	5.95E-04	1.64E-03
F19	2.19E-02	1.33E-02	4.30E-03	1.04E-03	1.83E-02
F20	1.94E-03	3.35E-03	3.35E-03	1.41E-03	4.22E-03
F21	2.66E-01	8.26E-02	1.45E-02	3.48E-02	2.66E-02
F22	1.00E+00	3.72E-05	3.21E-02	1.10E+01	5.58E-01
F23	2.14E-03	3.25E-04	2.12E-03	2.48E-02	2.74E-02

Table 8: Comparison of results on C1–C10

F	Metric	AFSSA-GSO	GSO	AFSSA-PSO	PSO	TSA	GWO	SSA
C1	AVG	**3.5453E+04**	6.6207E+05	1.4784E+05	4.2513E+05	7.2381E+07	5.4417E+06	2.2632E+06
	STD	**6.4145E+03**	2.3072E+04	2.6935E+04	4.1242E+05	3.4465E+06	7.4327E+05	1.2745E+06
C2	AVG	**1.7313E+01**	1.7869E+01	1.7563E+01	1.8121E+01	1.7745E+01	1.7687E+01	1.73929E+01
	STD	**2.5346E-09**	2.6386E-03	2.7325E-02	2.5726-03	3.1393-04	2.8583E-03	1.6342E-03
C3	AVG	**1.2713E+0**1	1.2776E+01	1.2741E+01	1.2797E+01	1.2774E+01	1.2785E+01	1.276E+01
	STD	**2.2530E-08**	2.3620E-04	2.7928E-05	2.0472E-04	2.6367E-02	4.1413E-03	6.3790E-04
C4	AVG	2.3326E+01	4.6235E+01	5.1420E+01	8.2034E+01	5.4418E+00	**4.2587E+00**	2.7348E+01
	STD	9.4127E+00	4.7238E+01	1.6832E+01	5.4971E+01	4.6345E+00	**3.2533E+00**	4.4701E+01
C5	AVG	**1.0124E+00**	1.1302E+00	1.0915E+00	5.5884E+00	1.0837E+00	1.0374E+00	1.0729E+00
	STD	**2.3427E-02**	3.5372E-01	4.6924E-01	6.8031E-01	8.4172E-01	2.2385E-01	2.2847E-01
C6	AVG	**1.0004E+00**	1.0675E+00	1.0126E+00	1.6384E+00	1.0060E+00	10417E+00	2.0471E+00
	STD	**1.2383E-01**	3.1764E+00	1.0765E+00	1.2417E+00	8.0114E-01	1.2371E+00	4.5361E+00
C7	AVG	1.1406E+02	1.6052E+02	2.2243E+02	7.6829E+02	**9.6043E+01**	3.2489E+02	4.3351E+02
	STD	2.4051E+02	1.3381E+02	2.5152E+02	3.3849E+02	**8.2208E+01**	1.6115E+02	2.3268E+02
C8	AVG	**3.0093E+00**	4.6836E+00	4.3237E+00	5.5572E+00	5.0425E+00	4.6735E+00	4.175E+00
	STD	**9.8557E-02**	2.4752E-01	2.4715E-01	6.5743E-01	1.9275E-01	2.9326E-01	1.5715E-01
C9	AVG	**2.0706E+00**	2.6836E+00	3.045E+00	4.6963E+00	2.0985E+00	2.1722E+00	2.1769E+00
	STD	**1.9947E-01**	4.3865E-01	4.0175E-01	2.2075E-01	6.6924E-01	1.2718E-01	4.5246E-01
C10	AVG	**1.0945E+00**	1.5752E+01	2.6353E+01	2.4281E+01	2.1423E+00	2.3026E+00	2.7591E+01
	STD	**2.4487E-01**	4.9682E+00	6.6931E+00	2.3235E+00	1.5042E-01	1.3068E+00	1.6034E+00

Table 13: The P-value results from the T-test for CEC2019 benchmarks

AGSO-GSO					
$F_i(x^*)$	**GSO**	**PSO**	**TSA**	**GWO**	**SSA**
C1	3.12E-04	4.81E-05	1.00E+00	2.47E-04	4.98E-03
C2	5.73E-03	1.62E-05	2.58E-04	3.65E-04	5.32E-04
C3	1.21E-04	3.62E-05	1.09E-03	2.51E-05	8.11E-04
C4	3.62E-04	4.68E-05	3.96E-04	3.82E-05	3.63E-04
C5	2.51E-04	2.34E-04	2.74E-04	2.76E-04	2.02E-04
C6	3.44E-04	1.08E-04	1.85E-03	2.02E-05	1.12E-04
C7	3.68E-04	3.53E-04	3.57E-04	3.64E-04	2.77E-04
C8	4.49E-04	4.74E-04	2.91E-04	2.12E-03	2.98E-04
C9	7.21E-04	3.35E-05	2.69E-04	1.12E+04	3.88E-04
C10	3.77E-03	3.56E-04	3.52E-04	3.59E-04	3.34E-04

AGSO-PSO					
$F_i(x^*)$	**GSO**	**PSO**	**TSA**	**GWO**	**SSA**
C1	7.09E-02	4.56E-05	1.39E-02	3.85E-03	2.82E-02
C2	3.14E-03	2.98E-05	3.58E-04	5.78E-03	2.43E-05
C3	3.06E-03	1.63E-03	2.83E-03	2.16E-03	2.64E-03
C4	1.57E-03	1.07E-05	1.65E-03	2.94E-03	1.31E-02
C5	6.85E-02	7.34E-03	4.12E-03	5.87E-04	1.71E-03
C6	2.32E-02	1.36E-02	4.20E-03	1.11E-03	1.89E-02
C7	2.08E-03	1.63E-03	3.47E-03	1.45E-03	2.04E-03
C8	2.72E-01	8.47E-02	1.38E-02	3.59E-02	2.73E-02
C9	1.00E+00	3.58E-05	3.28E-02	1.12E+01	5.72E-01
C10	2.21E-03	3.15E-04	2.18E-03	2.53E-02	2.81E-02

4.5 Data clustering

This section evaluates the performance of AFSSA in solving clustering problems. Given a dataset $S = \{x_1, \dots, x_N\}$ composed of N samples, each with D features, the goal is to partition S into K non-overlapping clusters $C = \{C_1, \dots, C_K\}$, such that $\bigcup_{j=1}^{K} C_j = S$ and $C_j \cap C_i = \emptyset$ for $j \neq i$. The clustering objective is to minimize the sum of squared intra-cluster distances defined by:

$$f(S, C) = \sum_{j=1}^{N} \min_{i \in \{1,\dots,K\}} \{|x_j - \mu_i\|^2\}, \tag{1}$$

where μ_i denotes the centroid of cluster C_i. The Euclidean distance metric is used to measure the distance between samples and cluster centroids, as given by:

$$d(x_i, x_j) = \sqrt{\sum_{m=1}^{D} (x_{im} - x_{jm})^2}. \tag{2}$$

Experimental setup and results

Although the number of clusters K and their general data characteristics are known in the benchmark datasets used, the main purpose of applying AFSSA is to demonstrate its effectiveness as a global optimization-based clustering method. The algorithm seeks to find cluster centroids that yield better compactness and separation compared to classical heuristics and swarm intelligence variants. This highlights AFSSA's potential as a reliable tool in practical clustering tasks where the data structure may be complex or high-dimensional.

AFSSA's clustering performance was evaluated on several widely used benchmark datasets summarized in Table 14, with fixed population size $N = 80$ and iteration count set to 1000 for all compared algorithms including GSO, PSO, TSA, GWO, and SSA.

Table 15 reports the mean and standard deviation (STD) of the objective function values (within-cluster sum of squared distances) over multiple independent runs. The mean indicates the average clustering quality achieved (lower is better), while the standard deviation reflects the consistency and reliability of the algorithm.

The results show that both AFSSA-GSO and AFSSA-PSO improve upon their original counterparts as well as other optimizers, achieving lower means and smaller standard deviations in most datasets. These observations suggest that AFSSA enhances the accuracy and stability of clustering solutions. Additionally, convergence analyses (Figure 4) demonstrate that AFSSA converges faster and more reliably, confirming its superiority in balancing exploration and exploitation for data clustering tasks.

Table 14: Clustering dataset

Data Set	**Instance Number**	**Number of Clusters**	**Number of Features**
Iris	150	3	4
Wine	178	3	13
CMC	1473	3	9
Glass	214	6	9
Vowel	871	6	3
Cancer	683	2	9

Table 15: Clustering objective function results

Dataset	**Metric**	**AFSSA-GSO**	**GSO**	**AFSSA-PSO**	**PSO**	**TSA**	**GWO**	**SSA**
Iris	Mean	98.28	103.46	103.85	114.14	104.5	112.25	102.25
	STD	4.02	3.98	3.48	7.26	3.85	7.55	2.55
Wine	Mean	16309.59	16414.06	16396.28	16672.55	16549.91	16881.29	16467.29
	STD	3.88	87.57	69.89	102.45	75.53	303.85	73.85
CMC	Mean	5581.72	5615.24	5596.81	6198.16	6343.16	6216.69	5695.69
	STD	19	28.41	23.0	231.77	49.03	105.88	81.88
Cancer	Mean	2979.34	3122.75	2988.37	3309.9	3269.02	3487.62	3154.62
	STD	7.71	93.23	17.14	141.22	83.89	118.75	78.75
Glass	Mean	247.23	248.39	247.2	275.73	316.6	343.47	281.47
	STD	1.57	1.46	2.32	16.87	23.91	35.12	24.49
Vowel	Mean	149884.36	153345.31	151233.75	158557.79	150754.03	155524.99	153550.99
	STD	255.07	303.25	361.09	854.1	286.55	332.25	314.25

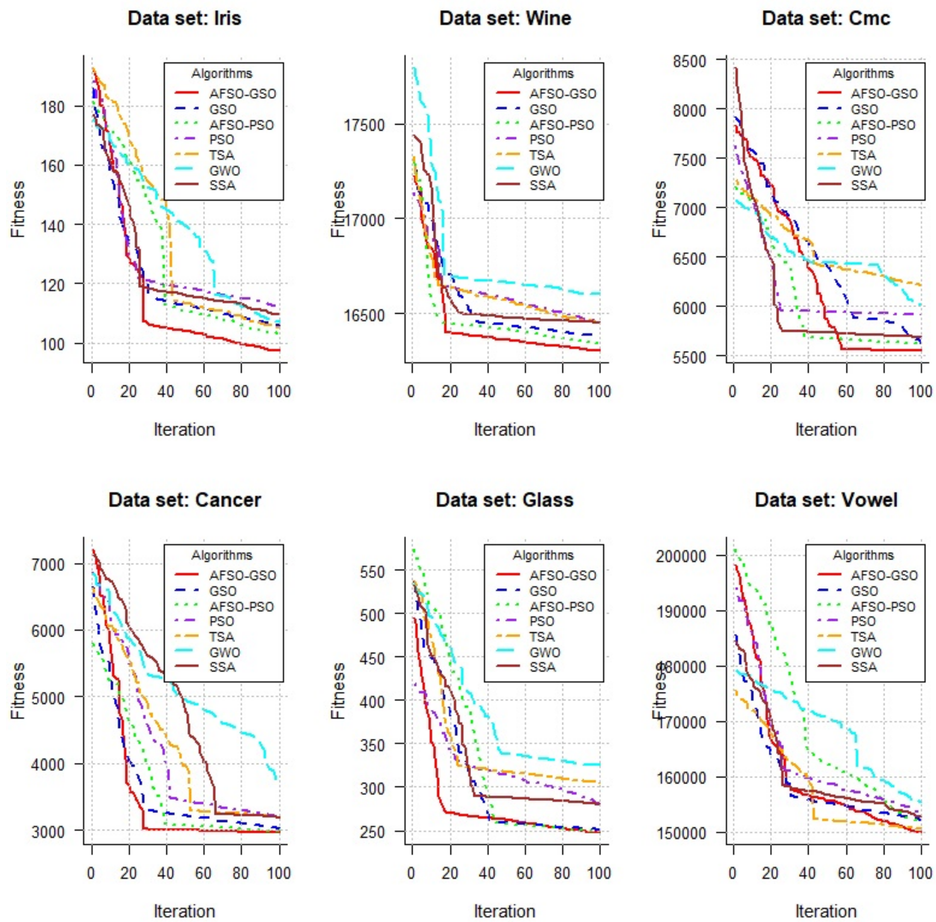

Figure 4: Convergences curve of the data clustering problem

5 Conclusion

This paper presented the Adaptive Fuzzy Swarm-based Search Algorithm (AFSSA), a novel control framework designed to enhance the adaptability of swarm intelligence algorithms in complex optimization problems. By integrating a Mamdani-type fuzzy inference system, AFSSA dynamically adjusts optimization coefficients during the search process, enabling smooth transitions between exploration and exploitation phases. This adaptive mechanism allows algorithms to automatically tailor their behavior to problem-specific characteristics, addressing a key limitation of conventional swarm-based methods in high-dimensional spaces. The effectiveness of AFSSA was validated through comprehensive experiments. When applied to PSO and GSO (resulting in AFSSA-PSO and AFSSA-GSO variants), the framework demonstrated consistent performance across 23 standard benchmark functions (30D to 500D) and the CEC2019 test suite. Additional validation on data clustering problems further supported its practical utility, suggesting broader applicability beyond numerical optimization. While these results are encouraging, it is acknowledged that performance may vary depending on problem structure and initialization conditions. Future research directions include optimizing the fuzzy rule base specifically tailored for different classes of optimization problems, allowing the inference system to better capture problem-specific dynamics and improve adaptation precision. Extending the AFSSA framework to other metaheuristic algorithms—including hybrid and multi-objective optimization methods can enhance its applicability to a wider array of complex scenarios. Furthermore, rigorous theoretical analysis of the convergence properties of AFSSA is essential to establish formal guarantees and deepen understanding of its behavior. These advancements aim to improve the robustness and computational efficiency of AFSSA, facilitating its broader adoption for challenging engineering and real-world optimization tasks.

Acknowledgement

The authors wish to express their appreciation for several excellent suggestions for improvements in this paper made by the referees.

References

[1] B. Abdollahzadeh, N. Khodadadi, S. Barshandeh, P. Trojovskỳ, F. S. Gharehchopogh, E. S. M. El-Kenawy, L. Abualigah, S. Mirjalili, *Puma optimizer (PO): A novel metaheuristic optimization algorithm and its application in machine learning*, Cluster Computing, **27**(4) (2024), 5235-5283. https://doi.org/10.1007/s10586-023-04221-5

[2] R. H. Abiyev, M. Tunay, *Optimization of high-dimensional functions through hypercube evaluation*, Computational Intelligence and Neuroscience, **2015**(1) (2015), 967320. https://doi.org/10.1155/2015/967320

[3] S. Aine, R. Kumar, P. P. Chakrabarti, *Adaptive parameter control of evolutionary algorithms to improve quality-time trade-off*, Applied Soft Computing, **9**(2) (2009), 527-540. https://doi.org/10.1016/j.asoc.2008.07.001

[4] M. Dorigo, M. Birattari, T. Stutzle, *Ant colony optimization*, IEEE Computational Intelligence Magazine, **1**(4) (2006), 28-39. https://doi.org/10.1109/MCI.2006.329691

[5] R. Eberhart, J. Kennedy, *A new optimizer using particle swarm theory*, In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science (MHS'95), IEEE, (1995), 39-43. https://doi.org/10.1109/MHS.1995.494215

[6] M. A. Ebrahim Mohamed, S. A. Ward, M. F. El-Gohary, *Hybrid fuzzy logic-PI control with metaheuristic optimization for enhanced performance of high-penetration grid-connected PV systems*, Scientific Reports, **15** (2025), 24650. https://doi.org/10.1038/s41598-025-09336-w

[7] R. Etesami, M. Madadi, *Tighter bounds on the Gaussian q-function based on wild horse optimization algorithm*, Journal of Algorithms and Computational Technology, **19** (2025). https://doi.org/10.1177/17483026251315392

[8] R. Etesami, M. Madadi, F. Keynia, *A new improved fruit fly optimization algorithm based on particle swarm optimization algorithm for function optimization problems*, Journal of Mahani Mathematical Research Center, **13**(2) (2024). https://doi.org/10.22103/jmmr.2023.20538.1362

[9] R. Etesami, M. Madadi, F. Keynia, *Principal component Gaussian optimization for enhancing metaheuristic algorithms in high-dimensional problems*, International Journal of General Systems, **2025** (2025), 1-36. https://doi.org/10.1080/03081079.2025.2525253

[10] R. Etesami, M. Madadi, F. Keynia, A. Arabpour, *Gaussian combined arms algorithm: A novel meta-heuristic approach for solving engineering problems*, Evolutionary Intelligence, **18**(2) (2025), 1-36. https://doi.org/10.1007/s12065-025-01026-w

[11] M. Eusuff, K. Lansey, F. Pasha, *Shuffled frog-leaping algorithm: A memetic meta-heuristic for discrete optimization*, Engineering Optimization, **38**(2) (2006), 129-154. https://doi.org/10.1080/03052150500384759

[12] Z. W. Geem, J. H. Kim, G. V. Loganathan, *A new heuristic optimization algorithm: Harmony search*, Simulation, **76**(2) (2001), 60-68. https://doi.org/10.1177/003754970107600201?urlappend=3%Futm_source%3Dresearchgate

[13] C. Huang, Y. Li, X. Yao, *A survey of automatic parameter tuning methods for metaheuristics*, IEEE Transactions on Evolutionary Computation, **24**(2) (2019), 201-216. https://doi.org/10.1109/TEVC.2019.2921598

[14] K. Hussain, M. N. M. Salleh, S. Cheng, Y. Shi, *On the exploration and exploitation in popular swarm-based metaheuristic algorithms*, Neural Computing and Applications, **31**(11) (2019), 7665-7683. https://doi.org/10.1007/s00521-018-3592-0

[15] K. Hussain, M. N. M. Salleh, S. Cheng, Y. Shi, *Metaheuristic research: A comprehensive survey*, Artificial Intelligence Review, **52** (2019), 2191-2233. https://doi.org/10.1007/s10462-017-9605-z

[16] O. Ibrahim, M. J. A. Aziz, R. Ayop, A. T. Dahiru, W. Y. Low, M. H. Sulaiman, T. I. Amosa, *Fuzzy logic-based particle swarm optimization for integrated energy management system considering battery storage degradation*, Results in Engineering, **24** (2024), 102816. https://doi.org/10.1016/j.rineng.2024.102816

[17] Y. T. Juang, S. L. Tung, H. C. Chiu, *Adaptive fuzzy particle swarm optimization for global optimization of multi-modal functions*, Information Sciences, **181**(20) (2011), 4539-4549. https://doi.org/10.1016/j.ins.2010.11.025

[18] S. Kaur, L. K. Awasthi, A. L. Sangal, G. Dhiman, *Tunicate swarm algorithm: A new bio-inspired based meta-heuristic paradigm for global optimization*, Engineering Applications of Artificial Intelligence, **90** (2020), 103541. https://doi.org/10.1016/j.engappai.2020.103541

[19] G. Li, T. Zhang, C. Y. Tsai, L. Yao, Y. Lu, J. Tang, *Review of the metaheuristic algorithms in applications: Visual analysis based on bibliometrics (1994-2023)*, Expert Systems with Applications, **124857** (2024). https://doi.org/10.1016/j.eswa.2024.124857

[20] N. Mahmoudi, A. Majidi, M. Jamei, M. Jalali, S. Maroufpoor, J. Shiri, Z. M. Yaseen, *Mutating fuzzy logic model with various rigorous meta-heuristic algorithms for soil moisture content estimation*, Agricultural Water Management, **261** (2022), 107342. https://doi.org/10.1016/j.agwat.2021.107342

[21] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, *Salp swarm algorithm: A bio-inspired optimizer for engineering design problems*, Advances in Engineering Software, **114** (2017), 163-191. https://doi.org/10.1016/j.advengsoft.2017.07.002

[22] S. Mirjalili, S. M. Mirjalili, A. Lewis, *Grey wolf optimizer*, Advances in Engineering Software, **69** (2014), 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

[23] I. Naruei, F. Keynia, *Wild horse optimizer: A new meta-heuristic algorithm for solving engineering optimization problems*, Engineering with Computers, **38**(Suppl 4) (2022), 3025-3056. https://doi.org/10.1007/s00366-021-01438-z

[24] B. H. Nguyen, B. Xue, M. Zhang, *A survey on swarm intelligence approaches to feature selection in data mining*, Swarm and Evolutionary Computation, **54** (2020), 100663. https://doi.org/10.1016/j.swevo.2020.100663

[25] M. Noroozi, H. Mohammadi, E. Efatinasab, A. Lashgari, M. Eslami, B. Khan, *Golden search optimization algorithm*, IEEE Access, **10** (2022), 37515-37532. https://doi.org/10.1109/ACCESS.2022.3162853

[26] H. R. Patel, V. A. Shah, *Type-2 fuzzy logic applications designed for active parameter adaptation in metaheuristic algorithm for fuzzy fault-tolerant controller*, International Journal of Intelligent Computing and Cybernetics, **16**(2) (2023), 198-222. https://doi.org/10.1108/IJICC-01-2022-0011

[27] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, *GSA: A gravitational search algorithm*, Information Sciences, **179**(13) (2009), 2232-2248. https://doi.org/10.1016/j.ins.2009.03.004

[28] P. Sharma, S. Raju, *Metaheuristic optimization algorithms: A comprehensive overview and classification of benchmark test functions*, Soft Computing, **28**(4) (2024), 3123-3186. https://doi.org/10.1007/s00500-023-09276-5

[29] Y. Shi, R. C. Eberhart, *Fuzzy adaptive particle swarm optimization*, In: *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546)*, IEEE, **1** (2001), 101-106. https://doi.org/10.1109/CEC.2001.934377

[30] R. Storn, K. Price, *Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces*, Journal of Global Optimization, **11** (1997), 341-359. https://doi.org/10.1023/A:1008202821328

[31] A. Taieb, H. Salhi, A. Chaari, *Adaptive TS fuzzy MPC based on particle swarm optimization-cuckoo search algorithm*, ISA Transactions, **131** (2022), 598-609. https://doi.org/10.1016/j.isatra.2022.05.018

[32] M. Tunay, R. Abiyev, *Improved hypercube optimisation search algorithm for optimisation of high dimensional functions*, Mathematical Problems in Engineering, **2022**(1) (2022), 6872162. https://doi.org/10.1155/2022/6872162

[33] D. Wang, Z. Yuan, A. Liu, Q. Lin, J. Qiao, *Model-free neuro-fuzzy q-learning control with swarm intelligence*, IEEE Transactions on Fuzzy Systems, (2025). https://doi.org/10.1109/TFUZZ.2025.3581421

[34] G. Xu, *An adaptive parameter tuning of particle swarm optimization algorithm*, Applied Mathematics and Computation, **219**(9) (2013), 4560-4569. https://doi.org/10.1016/j.amc.2012.10.067

[35] X. S. Yang, *A new metaheuristic bat-inspired algorithm*, In: Nature Inspired Cooperative Strategies for Optimization (NICSO 2010), Springer, (2010), 65-74. https://doi.org/10.1007/978-3-642-12538-6_6

[36] X. S. Yang, S. Deb, *Cuckoo search: Recent advances and applications*, Neural Computing and Applications, **24** (2014), 169-174. https://doi.org/10.1007/s00521-013-1367-1