

## Article

# Sensor Node Deployment Optimization for Continuous Coverage in WSNs

Haris Muhammad  and Haewoon Nam \* 

Department of Electrical Engineering, Hanyang University, Ansan 15588, Republic of Korea;  
haris737@hanyang.ac.kr

\* Correspondence: hnam@hanyang.ac.kr

**Abstract:** Optimizing sensor node coverage remains a central challenge in wireless sensor networks (WSNs), where premature convergence and suboptimal solutions in traditional optimization methods often lead to coverage gaps and uneven node distribution. To address these issues, this paper presents a novel velocity-scaled adaptive search factor particle swarm optimization (VASF-PSO) algorithm that integrates dynamic mechanisms to enhance population diversity, guide the search process more effectively, and reduce uncovered areas. The proposed algorithm is evaluated through extensive simulations across multiple WSN deployment scenarios with varying node densities, sensing ranges, and monitoring area sizes. Comparative results demonstrate that the approach consistently outperforms several widely used metaheuristic algorithms, achieving faster convergence, better global exploration, and significantly improved coverage performance. On average, the proposed method yields up to 14.71% higher coverage rates than baseline techniques. These findings underscore the algorithm's robustness and suitability for efficient and scalable WSN deployments.

**Keywords:** particle swarm optimization (PSO); adaptive search factor; fast convergence; wireless sensor network; Delaunay triangulation



Academic Editor: Carles Gomez

Received: 19 April 2025

Revised: 3 June 2025

Accepted: 3 June 2025

Published: 9 June 2025

**Citation:** Muhammad, H.; Nam, H. Sensor Node Deployment Optimization for Continuous Coverage in WSNs. *Sensors* **2025**, *25*, 3620. <https://doi.org/10.3390/s25123620>

**Copyright:** © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Wireless sensor networks (WSNs), empowered by compact micro-electromechanical system (MEMS) technologies, are widely used in applications such as environmental monitoring and disaster management. These networks face persistent challenges, including energy constraints and coverage optimization, especially in dynamic and inaccessible environments [1]. Unlike traditional networks, WSNs operate autonomously without fixed infrastructure and are specifically designed to monitor and communicate various physical or environmental parameters, including temperature, motion, noise, pressure, vibration, and pollution [2]. These networks rely on a central entity commonly referred to as a sink or base station that acts as a bridge between users and the network, enabling data access and query functionality. A typical WSN comprises hundreds or even thousands of sensor nodes, requiring advanced and adaptive protocol designs to effectively operate within dynamic environments and limited resource conditions [3]. Over the past decade, there has been a significant surge in research, standardization efforts, and industrial investment in WSN technologies [4]. WSNs are now integral to a wide range of applications, particularly in scenarios where human access is limited, such as environmental monitoring, disaster recovery, pollution detection, military surveillance, healthcare systems, and smart home environments [5].

Each sensor node in a WSN is capable of independently collecting, storing, processing, and exchanging environmental data with neighboring nodes [6]. However, coverage gaps in areas not monitored by any sensor can emerge when regions fall outside the sensing range. To address this, researchers have proposed integrating mobile nodes with energy-efficient metaheuristics like the scalable firefly algorithm, which enhances coverage by dividing the network into manageable cells and enabling accurate estimation of physical coverage [7]. Additionally, hybrid optimization frameworks, such as those combining branch-and-cut algorithms with reinforcement learning, have been developed to ensure comprehensive coverage. To further improve network performance and mitigate interference, effective strategies for beamforming and re-optimization are crucial [8]. A notable contribution in this area is the wireless sensor network hole detection (WHD) technique, which quantifies the presence of coverage holes in randomly deployed sensor fields within regions of interest (ROIs) [9]. This method incorporates metrics such as average hole detection time to balance energy efficiency with quality of service (QoS) objectives [10].

Current research primarily focuses on developing computationally and energy-efficient algorithms and protocols, particularly for monitoring and control applications [11]. Despite their capabilities, sensor nodes are constrained by limited battery life, processing power, and memory for data transmission and detection [12]. The accuracy and comprehensiveness of the data collected improve when sensors are positioned closer to the observed events. Since their introduction, WSNs have found extensive use in security and surveillance, as well as in industries like factory automation, military operations, remote patient monitoring, environmental observation, and disaster prediction [13–17]. Sensor nodes play a critical role in these applications, gathering essential environmental data and transmitting it to the base station (BS) or sink via multicast communication [18]. However, the network's structure may increase energy consumption in nearby nodes, potentially causing energy depletion or “holes” within the network [19]. Generally, WSNs consist of numerous sensor nodes transmitting environmental data to a base station for aggregation and analysis.

Coverage maintenance remains a key challenge due to battery limitations and node failures. Several lightweight distributed methods have been proposed to detect and address coverage gaps using mobile nodes, though many incur significant communication overhead [20,21]. In WSNs, a two-step process is typically employed to maintain coverage. First, a lightweight distributed system enables nodes to assess their environment and identify coverage gaps [22]. Once the number of gaps exceeds a certain threshold, the algorithm selects and prioritizes mobile nodes to fill these gaps, focusing on those closest to the gaps and with the highest priority. Over the past decade, advancements in embedded micro-sensing and wireless communication technologies have led to the development of increasingly compact and affordable wireless devices [23]. Several techniques leverage nodal attributes related to network topology and geometric connectivity to detect coverage gaps. However, most algorithms require a high average node degree to identify specific patterns, resulting in significant communication overhead [24]. Notably, while these methods can identify nodes near coverage gaps, they do not provide efficient clustering mechanisms, which could simplify coverage maintenance. This high communication cost, unfortunately, shortens the lifespan of WSNs considerably. Similarly, our PSO-based approach enhances sensor placement and coverage in WSNs to improve overall network performance. Recent research on active reconfigurable intelligent surface (RIS)-aided non-orthogonal multiple access (NOMA)-enabled space–air–ground integrated network (SAGIN) with UAVs highlights advanced energy-efficient optimization strategies [25]. Similarly, our PSO-based approach enhances sensor placement and coverage in WSNs to improve overall network performance.

Extending network lifetime is challenging due to limited data speed and the difficulty of replacing or recharging batteries in randomly dispersed sensor nodes (SNs), often in inaccessible areas [26]. Moreover, energy consumption increases more than linearly with communication distance, underscoring the critical need for efficient energy management strategies. In multi-hop communication systems, sensor nodes located near the sink experience disproportionately heavy traffic loads, causing their energy to deplete faster—this phenomenon is commonly referred to as the “Energy Sink-Holes Problem” or “Bottleneck Problem” [27,28]. To address coverage optimization, a perception model combining path loss and false alarm probability has been developed [29]. Furthermore, a novel multi-objective optimization framework has been proposed for heterogeneous-function WSNs, balancing coverage, connectivity, and cost. This approach introduces an additional cost dimension to accommodate the distinct monitoring requirements of various functional sensors within specific areas. By employing an efficient node deployment strategy, redundancy and costs are reduced while simultaneously improving network coverage, robustness, energy efficiency, and load balancing, ultimately contributing to an extended network lifetime [30]. To maximize coverage, minimize the number of deployed nodes, and maintain network connectivity, the proposed method is based on a novel optimization approach called the Nutcracker Optimization Algorithm (NOA) [31]. Reliable Clustering with Optimized Scheduling and Routing for Wireless Sensor Networks introduces a novel GridCosins Chain Clustering technique that uses GridCosins distance for node grouping and forms distance-based tree topologies, reducing transmission range and enhancing network lifetime [32].

Ensuring complete coverage in WSNs is a complex task that has attracted considerable research focused on developing effective solutions to close coverage gaps, improve network efficiency, and enhance overall performance. As a result, there have been many recent research developments. Multi-objective coverage algorithms for WSNs that optimize energy-saving and coverage rate metrics are noteworthy advances [33]. The whale optimization algorithm (WOA), which draws inspiration from the hunting strategies of whales, has also been used to optimize WSNs [34]. Reverse learning has been included to increase network coverage. Furthermore, it has been suggested that Lévy flights be included in WOA to guarantee extensive network coverage. The bee algorithm, particle swarm optimization algorithm, weed algorithm, wolf pack algorithm, glowworm swarm optimization, simulated annealing (SA), ant colony optimization (ACO), improved grey wolf optimizer with multi-strategies, improved cuckoo search, and social spider optimization (SSO) are some of the other intelligent optimization algorithms that have been investigated. Together, these various algorithms help to advance coverage tactics in WSNs.

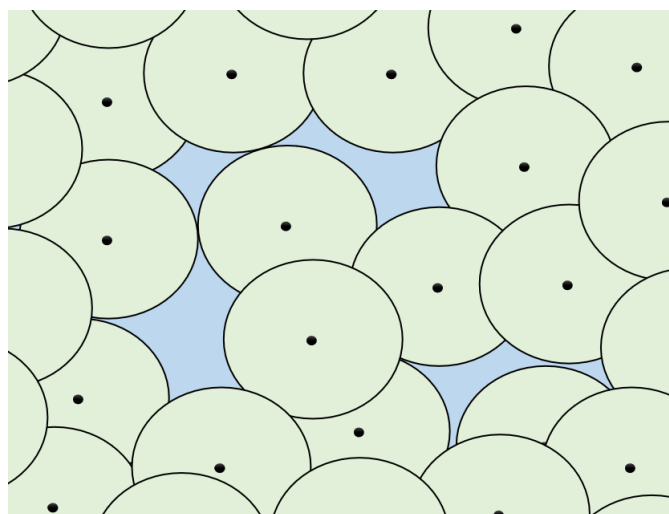
Recent research has demonstrated the effectiveness of PSO-based techniques in WSN optimization, particularly in enhancing localization accuracy through error minimization and sensor placement optimization [35]. In reconfigurable WSNs (RWSNs), PSO-based algorithms address complex multi-peak optimization problems, improving sensor deployment efficiency and coverage quality [36]. Self-adaptive estimation PSO (SEPSO) further refines sensor placement by reducing computation time compared to traditional methods like the virtual force approach [37]. In heterogeneous WSNs (HWSNs), PSO-based methods enhance energy efficiency and convergence speed while minimizing redundancy and coverage gaps [38]. Deploying sensors in 3D WSNs presents additional challenges due to higher deployment costs and connectivity constraints, where PSO variants such as virtual force-directed PSO (VFPSO) have been proposed to maximize coverage while minimizing energy consumption [39]. Moreover, integrating PSO with variable domain chaos optimization (VDCOA) has been shown to improve network efficiency, sensor distribution, and coverage rates [40]. In WSN clustering, PSO has been utilized for optimal cluster head

selection by leveraging hybrid PSO and K-means clustering to enhance network lifespan and balance energy consumption [41].

Particle swarm optimization, developed over the past two decades, offers significant advantages over previous intelligent optimization techniques, including simpler iteration rules and faster convergence rates [42]. This optimization algorithm has been effectively implemented across various engineering domains. For instance, co-evolutionary PSO algorithms have been employed to explore dynamic deployment optimization in WSNs [43]. Optimization algorithms play a crucial role in enhancing WSN's performance. The hybridization of the global PSO algorithm with shuffled frog leaping optimization has resulted in a more effective problem-solving approach [44]. Furthermore, hybrid algorithms integrating PSO with Genetic Algorithms have been developed to maximize WSN coverage. At the same time, a combination of PSO and simulated annealing has been proposed for energy-efficient coverage optimization [45].

### Motivation

WSNs are widely used in applications that require reliable and high-quality coverage, such as environmental monitoring, surveillance, and industrial automation. However, achieving uniform sensor distribution and complete area coverage remains a significant challenge. Factors like irregular node placement, limited energy resources, and inefficient deployment strategies often result in coverage gaps and uneven sensing performance. Many existing optimization methods, particularly those based on PSO, struggle with issues such as premature convergence and getting trapped in local optima. These limitations can lead to poor sensor placements and large uncovered areas, which in turn disrupt both sensing and communication, as illustrated by the light blue-highlighted region in Figure 1.



**Figure 1.** Coverage hole problem in WSNs.

To address these challenges, this paper proposes a velocity-scaled adaptive search factor PSO algorithm, which improves the global search ability and speeds up convergence. The algorithm begins with Hammersley sequence-based initialization to ensure an even initial distribution of nodes, and it uses Delaunay triangulation to effectively detect and resolve coverage holes. The main goal of this research is to offer a robust and efficient optimization technique that ensures maximum coverage, uniform sensor placement, and improved search accuracy. By overcoming the key drawbacks of earlier approaches, the proposed VASF-PSO significantly enhances WSN coverage performance and offers a practical solution for real-world deployments.

## 2. Related Work

A comprehensive approach integrates deployment, detection, and network models to address coverage gaps with varying characteristics. The framework aims to enhance coverage quality, extend network lifespan, and reduce sensor usage under uncertain parameters [46]. It employs planning, selection, movement, and decision-making techniques, categorized into distribution, centralization, stochastic, heuristic, approximation, and classical algorithms [47]. An improved particle swarm optimization algorithm (NS-IPSO) is also introduced to efficiently segment sensor nodes by enhancing distance estimation accuracy between unknown and anchor nodes, enabling better regional division. This approach identifies sensor nodes with shorter paths to anchor nodes, ensuring optimal placement. A surveillance network monitors partial discharges in high-voltage equipment using ultrasonic sensors designed for 3D alignment. A secure sensor cooperation strategy enhances target acquisition, reduces delays, and saves energy [48]. Advanced algorithms, including Gaussian and polynomial kernel integration and graph-based manifold learning, enable precise localization by mapping physical spaces to high-dimensional data. Their performance has been validated under various conditions. A distributed self-healing mechanism also detects and closes coverage gaps, accurately predicting their size and location. Furthermore, by predicting the amount and location of coverage gaps, a distributed self-healing system automatically finds and fixes them, improving the resilience of networks without the need for centralized control.

WSNs optimize coverage and minimize energy use by coordinating node movements to fill gaps without disrupting connectivity. Sensor nodes share maintenance data with the cluster head (CH), selected based on cooperation rank, transmission rank, and residual energy [49]. Mobile nodes gather detection data via intermediary channels, reducing latency and energy consumption. A flexible, energy-efficient protocol addresses QoS challenges by accommodating sensor resource constraints. To estimate coverage gaps, a computational geometry method uses Delaunay triangulation and empty circle attributes to identify and assess gaps in large-scale networks [50].

### *Contributions*

This paper proposes the velocity-scaled adaptive search factor PSO algorithm, which dynamically adjusts the search factor based on the distance of a particle from the swarm's optimal position, effectively balancing exploration and exploitation. In the PSO algorithm, the search factor controls the impact of the particle's previous velocity on its current velocity, thereby balancing local and global exploration of the search space. Local exploration focuses on finding the best solutions near the particle's current position, while global exploration seeks optimal solutions across the entire search space. A lower search factor facilitates local exploitation, while a higher search factor promotes global exploration.

The contributions of the paper are summarized as follows:

1. **Adaptive Algorithm:** This paper proposes a new PSO variant where the search factor is dynamically scaled based on particle velocity relative to the global best. Unlike conventional PSO or previous adaptive methods, our algorithm increases the exploration capability when particles are far from the optimal region and enhances exploitation as particles converge. This balance significantly improves convergence speed and solution quality.
2. **Delaunay triangulation:** Moreover, although Delaunay triangulation has been applied in computational geometry and certain network-related contexts, its integration into a PSO-based WSN coverage optimization framework is novel. This incorporation enables accurate detection and quantification of coverage holes, allowing the opti-



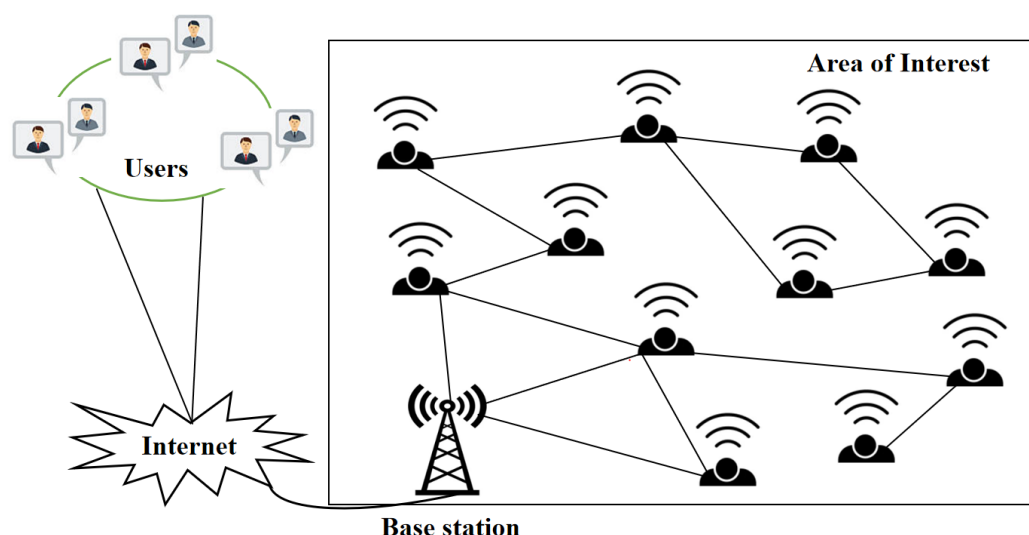
mization process to directly target practical performance metrics such as redundancy reduction and reliability enhancement in WSN deployments.

3. **Hammersley Sequence-Based Initialization:** We replace conventional random initialization with Hammersley sequence-based initialization (HSI) to ensure a well-distributed initial swarm, reducing the likelihood of premature convergence. Our integration of HSI into VASF-PSO demonstrates for the first time its measurable impact on WSN coverage efficiency.
4. **WSN Deployments:** The proposed VASF-PSO is evaluated not only on standard benchmark functions but also in real WSN deployment scenarios. We provide a thorough comparison with existing state-of-the-art algorithms (HPO, ABC, IGWO-MS, ICS, and standard PSO), showing superior performance in convergence speed, coverage rate, and node distribution uniformity.

### 3. System Model

#### 3.1. Coverage Problem in WSNs

WSNs encounter a coverage problem that is closely tied to the energy hole issue, where uneven energy consumption among sensor nodes leads to coverage gaps in the monitored area [51]. This happens because certain nodes, especially those near the sink or in high-traffic zones, consume energy faster than others. These nodes, which often relay data for other nodes, deplete their energy more quickly, leading to coverage holes when they stop functioning [52]. This disrupts the network's primary goal of reliable, continuous data collection. High energy consumption near the sink causes rapid depletion of nearby nodes, isolating distant nodes and reducing overall network performance. The imbalance in energy usage creates localized shortages, affecting coverage. Additionally, in sparse deployments, the failure of a single node can cause significant coverage gaps, while dense deployments may result in excessive energy consumption due to overlapping sensor ranges. These coverage gaps hinder monitoring, compromise data reliability, shorten network lifetime, and can even lead to network partitioning, where nodes beyond the gap lose connectivity to the sink, rendering them ineffective despite having remaining energy [53]. The structure of the WSN network is shown in Figure 2.



**Figure 2.** Structure of WSN system.

Holes are often unavoidable in wireless sensor networks, as they can arise from random node deployment, environmental factors, or external threats [54]. A hole refers to a region that becomes inaccessible or disconnected, either temporarily or permanently, due to

node failure, energy depletion, or other disruptions. Such coverage and connectivity gaps can severely impair the network's sensing and communication capabilities, particularly in critical applications. Prompt detection and remediation are therefore essential to ensure the network's performance and reliability.

### 3.2. Coverage Problem Optimization Formulation

Sensor nodes are assumed to be randomly distributed within a rectangular region of size  $L_x \times L_y$ , and  $N$  sensor nodes ( $N_1, N_2, N_3, \dots, N_n$ ) are randomly deployed. This random distribution is assumed to be uniform, meaning that the  $x$  and  $y$  coordinates of each sensor node are independently drawn from a continuous uniform distribution over the spatial dimensions of the deployment region, i.e.,  $x \sim \mathcal{U}(0, L_x)$  and  $y \sim \mathcal{U}(0, L_y)$ . As a result, each location within the rectangular area has an equal probability of being selected for node placement, ensuring an unbiased and spatially neutral initial deployment. Each node is assigned an equal coverage radius, denoted as  $r_c$ , where  $N_i = (x_i, y_i), i = 1, 2, 3, \dots, n$ . The research aims to maximize the network's lifespan while ensuring that the specified coverage requirements have been met by carefully arranging sensor nodes around the region. Following the splitting of the monitoring area into multiple parts, each subregion's coordinate location is obtained as  $R_j = (x_j, y_j, j = 1, 2, 3, \dots, p * q)$ ; following that, the destination point's Euclidean distance from the sensor node may be expressed as

$$d(N_i, R_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, \quad (1)$$

This paper adopts a probabilistic sensing model to more precisely characterize the sensing performance of sensor nodes within the target region, effectively incorporating uncertainties arising from noise and environmental obstacles. In contrast to traditional binary models, where a sensor is assumed to either fully cover a point within a fixed radius or not at all, this model reflects the gradual decline in sensing accuracy as distance from the sensor increases. The sensing probability within the monitored region is thus defined as follows:

$$P(N_i, R_j) = \begin{cases} 0, & r_c + r_e \leq d(N_i, R_j) \\ e^{-\lambda \alpha^\beta}, & r_c - r_e < d(N_i, R_j) < R_s + R_u \\ 1, & r_c - r_e \geq d(N_i, R_j), \end{cases} \quad (2)$$

where  $r_c$  represents the core sensing radius within which sensing is nearly certain, while  $r_e$  denotes an extended sensing range beyond the core radius where sensing probability decays exponentially. The term  $\alpha = r_e - r_c + d(N_i, R_j)$  models the effect of increasing distance on sensing probability. Parameters  $\lambda$ ,  $\beta$ , and  $\gamma$  control the rate and shape of this exponential decay, enabling the model to adapt to varying sensor characteristics and environmental conditions.

The total sensing probability, if more than one sensor node can locate the target location, is as follows:

$$P(N, R_j) = 1 - \prod_{i=1}^n (1 - P(N_i, R_j)), \quad (3)$$

where  $P(N, R_j)$  is the total sensing probability,  $n$  represents the total number of sensor nodes, and  $N$  denotes the set containing all sensors. The objective function for the WSN coverage optimization problem is designed to identify an effective node deployment strategy that maximizes coverage. The coverage rate of a WSN can be characterized as the ratio of the number of targets covered by the sensor set to the total number of targets in the area, expressed as

$$A_{\text{cov}} = \frac{\sum_{j=1}^{L_x \times L_y} P(N_i, R_j)}{L_x \times L_y}. \quad (4)$$

where  $A_{\text{cov}}$  represents the coverage rate of the area and is used as the objective function in the WSN coverage optimization.

### 3.3. Conventional Optimization Algorithms

The basic ideas of the conventional PSO algorithm are summarized in this part. In contrast, the following part contains in-depth descriptions of the PSO, ant colony optimization, and simulated annealing (SA) algorithms. In contrast to the conventional PSO, some of these algorithms have been specially modified to deal with the WSN coverage optimization issue. This work compares the basic PSO with other traditional optimization algorithms like ACO, SA, and adaptive particle swarm optimization (APSO), in addition to employing it as the main algorithm. This study emphasizes the benefits of PSO in the context of WSN coverage optimization. PSO is favored for its ability to adapt and be efficient in resolving difficult optimization issues.

#### Traditional PSO

The classic PSO algorithm is a population-based optimization algorithm that draws inspiration from the collective behavior of flocks of birds or schools of fish [55]. A potential solution is represented by each particle in the swarm, which travels around the solution space according to its velocity and the optimal locations discovered by other particles. Since its initial development, PSO has been expanded to address a variety of issues, including discrete, multi-objective, and restricted optimization. It is widely used in fields including biology, engineering, and finance for clustering, parameter estimation, and optimization because of its ease of use and effectiveness. With particles interacting to discover the optimal solution, PSO's swarm intelligence enables quick exploration and exploitation of the solution space. To ensure quick convergence to the optimum solution, the velocity update method balances exploring the search space with making use of the most prominent position.

PSO iteratively updates the location and velocity of each particle. The velocity modifies the particle's location by directing movement inside the search space. Until a stopping requirement is satisfied, like a suitable solution, these updates keep coming. In an  $N$ -dimensional optimization problem, the velocity and location of the  $i$ th particle are expressed as  $V_{it}^k = (v_1, v_2, \dots, v_{itN})$  and  $X_{it}^k = (x_1, x_2, \dots, x_{itN})$ . The position and velocity of particle  $i$  in a swarm of  $N$  particles are updated as

$$V_{it}^k = \omega_k V_{it}^{k-1} + c_1 r_1 (P_{\text{best},it} - X_i^k) + c_2 r_2 (G_{\text{best}} - X_i^k), \quad (5)$$

$$X_{it}^k = X_{it}^{k-1} + V_{it}^k. \quad (6)$$

The inertial weight is represented by  $\omega_k$ , the optimal location determined by the swarm as a whole is  $G_{\text{best}}$ , whereas  $P_{\text{best},it}$  denotes the individual best position for each particle. While  $r_1$  and  $r_2$  are random variables falling within the range of  $[0, 1]$ , the acceleration coefficients  $c_1$  and  $c_2$  represent the personal and global components, respectively. Furthermore, the particle's updated velocity is  $V_{it}^k$ , and its current velocity is  $V_{it}^{k-1}$ . Whereas  $X_{it}^k$  denotes the  $i$ th particle's prior position,  $X_{it}^k$  denotes its updated position. In PSO, the acceleration components  $c_1$  and  $c_2$  regulate the exploration and exploitation of the search space [56]. The cognitive component,  $c_1$ , determines the impact of the particle's personal best location  $P_{\text{best},it}$  on its velocity update. A larger value of  $c_1$  leads to a greater influence, pushing the particle to explore its own best. The social component,  $c_2$ , determines the



effect of the global best position  $G_{best}$  on the particle's velocity. A higher  $c_2$  value drives the particle to explore the best position discovered by the swarm. The acceleration components  $c_1$  and  $c_2$  are updated as

$$c_1^{(k)} = (c_{1f} - c_{1i}) \times \frac{i}{iter_{max}} + c_{1i}, \quad (7)$$

$$c_2^{(k)} = (c_{2f} - c_{2i}) \times \frac{i}{iter_{max}} + c_{2i}, \quad (8)$$

where the term  $c_1^{(k)}$  represents the cognitive acceleration coefficient,  $c_2^{(k)}$  denotes the social acceleration coefficient,  $c_i$  and  $c_f$  refer to the initial and final limits of the acceleration coefficients, respectively.

## 4. Proposed Optimization Algorithm

The PSO algorithm has successfully addressed several deployment issues in WSNs, but improving its fundamental concepts is crucial to its success. Overcoming the drawbacks of the simple PSO algorithm and incorporating more sophisticated algorithms are crucial for achieving better performance in WSN deployment. Due to the dynamic nature of WSN settings, these enhancements can increase convergence speed, resilience, and adaptability while also improving sensor location and energy economy.

### 4.1. Optimization Search Factor

In swarm intelligence (SI) algorithms, such as PSO, finding an appropriate balance between exploration and exploitation is vital. In the PSO algorithm, the balance parameter between exploration and exploitation is the search factor  $\omega_k$ . As  $\omega_k$  changes dynamically throughout the optimization process, it regulates how much the particle's past velocity affects its present velocity. A high search factor enables particles to explore a larger portion of the search space, promoting diversity and broad exploration. As the process progresses, particles focus on refining their positions near the best-known solution by reducing the value of  $\omega_k$  to help exploitation, and the search factor is derived as

$$\omega_k = \omega_{min} + (\omega_{max} - \omega_{min}) \left( 1 - \frac{|V_{it}^{k-1}|}{V_{max}} \right). \quad (9)$$

where  $V_{max}$  is the maximum allowable velocity for the particle. If the velocity exceeds this value, it is typically clipped to  $V_{max}$  to prevent overlapping behavior in the search space. As the particle's velocity increases, the search factor  $\omega_k$  decreases.  $\omega_{min}$  and  $\omega_{max}$  are the minimum and maximum limits of the search factor. The velocity-scaled adaptive search factor  $\omega_k$  in (9) dynamically adjusts the search factor of each particle according to its velocity magnitude. This approach is motivated by the need to balance exploration and exploitation during the search process. When a particle's velocity  $|V_{it}^{k-1}|$  is low, it indicates that the particle is potentially near a promising solution region or a local optimum. In this case, increasing the search factor toward  $\omega_{max}$  encourages the particle to explore more broadly and avoid premature convergence. Conversely, when the velocity is high, the search factor is reduced toward  $\omega_{min}$ , focusing the search on exploitation to refine the solution around the current position. By normalizing the velocity by its maximum value  $V_{max}$ , the search factor is adaptively scaled in each iteration and for each particle, improving convergence speed and preventing stagnation. As the optimization progresses and particle velocities decrease, indicating a trend toward convergence, the adaptive factor reduces the influence of randomness. This results in more refined, localized search behavior, which is essential for fine-tuning node positions to maximize coverage continuity.

and eliminate minor overlaps and gaps. Like a nonlinear convergence factor, this adaptive mechanism balances exploration and exploitation, facilitating rapid space exploration and precise solution refinement. The pseudo-code of the proposed VASF-PSO algorithm is demonstrated in Algorithm 1.

---

**Algorithm 1** VASF-PSO pseudocode.
 

---

```

1: Input: Initialize swarm size, particle velocities ( $V_i^k$ ), positions ( $X_i^k$ ), search factor ( $w_k$ ),
   individual best ( $P_{best_i}$ ), swarm best ( $G_{best}$ ), and iteration count.
2: Output: Best fitness
3: Initialize particles using Equation (10)
4: Locate the coverage holes using Equations (11)–(13)
5: for  $k = 1$  to Max Iterations do
6:   for  $i = 1$  to swarm size do
7:     Calculate the fitness using Equation (4).
8:     if fitness <  $P_{best_i}$  then
9:       Update  $P_{best_i}$  to the current fitness value.
10:    end if
11:    if fitness <  $G_{best}$  then
12:      Update  $G_{best}$ .
13:    end if
14:    Update the velocity of each particle using Equation (5).
15:    Update the position of each particle using Equation (6).
16:    Calculate acceleration coefficients using Equations (7) and (8)
17:    Update the search factor for VASF-PSO using Equation (9)
18:    Repeat steps 1 to 18 until the maximum number of iterations is reached.
19:  end for
20: end for
21: Return:  $G_{best}$ , Optimized continuous WSNs deployment.
  
```

---

To achieve optimal results across various testing configurations, the parameters of the VASF-PSO algorithm are carefully tuned based on empirical testing and sensitivity analysis. The maximum number of iterations  $k$  is set to 100, and the population size  $it$  is selected at 30, falling within the typical range of 20 to 50 particles, considering the problem's complexity and dimensionality. A significant contribution of the proposed VASF-PSO algorithm is the dynamic adjustment of the search factor  $\omega_k$ , which ranges from  $\omega_{\min} = 0.2$  to  $\omega_{\max} = 0.9$ . This range was established through extensive empirical testing across various benchmark functions, ensuring stable convergence and a balanced approach between exploration and exploitation in diverse problem landscapes. The cognitive coefficient  $c_1$  and social coefficient  $c_2$  are varied using time-varying acceleration coefficients (TVAC), with  $c_1$  decreasing from 2.5 to 0.5 and  $c_2$  increasing from 0.5 to 2.5. Initially, a higher  $c_1$  fosters exploration, while a lower  $c_2$  reduces the influence of the global best, guiding the search toward the global optimum as iterations progress [57]. To enhance diversity, random numbers  $r_1$  and  $r_2$  are initialized within  $[0, 1]$ , and the velocity range is constrained to  $[-2, 2]$ . A moderate value for the velocity limit  $V_{\max}$  was empirically selected to facilitate smooth convergence during the optimization process. This careful choice ensures that particles approach optimal solutions gradually, minimizing the risk of overshooting and maintaining the integrity of the search. The search factor range (0.2–0.9) and other parameters in Table 1 were selected based on the standard PSO procedure and validated by initial testing, demonstrating consistent and dependable performance. A comprehensive sensitivity analysis will be part of future work to evaluate how these factors affect the algorithm's resilience and performance.

**Table 1.** VASF-PSO algorithm parameter selection.

Parameter	Symbol	Value
Population size	$it$	30
Maximum iterations	$k$	100
Search factor	$\omega_k$	0.2–0.9 (Adaptively adjusted by VASF-PSO)
Cognitive coefficient	$c_1$	Decreases from 2.5 to 0.5
Social coefficient	$c_2$	Increases from 0.5 to 2.5
Velocity range	$V_{\max}$	$[-2, 2]$
Random numbers	$r_1, r_2$	Random values in $[0, 1]$

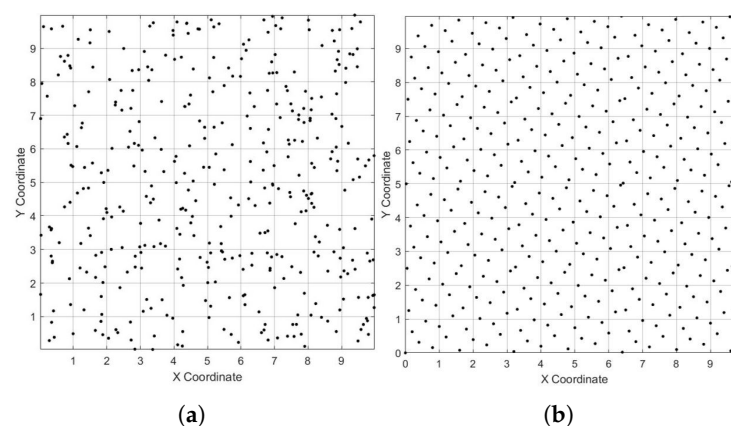
#### 4.2. Hammersley Sequence-Based Initialization

The Hammersley sequence is a method for generating a set of quasi-random locations that may be used to deploy sensor nodes in WSNs. The purpose of the first sensor node deployment is to guarantee that nodes are distributed uniformly and non-clustered over a region, hence improving network coverage and performance. By distributing sensor nodes evenly throughout the deployment region, the Hammersley sequence avoids gaps or clustering. By reducing the disparity between points, its low-discrepancy behavior improves coverage while using fewer nodes. This leads to increased network coverage, which is necessary for a wireless sensor network to be dependable and effective.

A random distribution is used by the PSO algorithm to initialize the population within the search space. Nevertheless, this algorithm frequently leads to unpredictability, unequal individual distribution, and poor exploration capabilities, all of which may affect the algorithm's performance. The population was initialized using the Hammersley sequence to get around this limitation. To increase population diversity and speed up the convergence process, the Hammersley sequence is renowned for its uniform distribution qualities, quick convergence, and low computing cost. Particles are dispersed uniformly over the search space due to the Hammersley sequence, which is utilized to initialize the population with a more uniform distribution. The location of each particle  $k$  is computed by properly scaling the sequence values. A factor  $W$  is applied to the first coordinate  $x'_i$ , and  $H$  is applied to the second coordinate  $y_i$ , which results in using the Van der Corput sequence with base 2. This method is represented numerically as

$$H_s i = (x'_i, y'_i) = (H_s i \times W, \text{VanDerCorput}(i, 2) \times H). \quad (10)$$

$\text{VanDerCorput}(i, 2)$  refers to the Van der Corput sequence, which is generated using base 2 for the index  $i$ . The Van der Corput sequence is a low-discrepancy sequence, producing numbers that are more uniformly distributed compared to random numbers. In the case of WSNs, it ensures that the nodes are spaced more uniformly across the area, especially when dealing with a large number of nodes. The Hammersley sequence is a method for generating a set of quasi-random locations, and random initialization is shown in Figure 3.

**Figure 3.** Initial distribution of PSO particles. (a) Random initialization. (b) Hammersley sequence.

### 4.3. Convergence Hole Problem

In this research work, we address the convergence hole problem in WSNs using the Delaunay triangulation method. Delaunay triangulation is an algebraic method that splits a collection of points into independent triangles while ensuring that no point is inside the circumcircle of any triangle. This approach is frequently used in network design to improve coverage and connectivity. The network's coverage may be optimized via Delaunay triangulation, therefore reducing convergence holes. It ensures that the sensor nodes are evenly distributed across the network, uncovering the convergence hole and leading to more efficient and reliable network operation.

Delaunay triangulation is crucial for identifying coverage holes and network connectivity issues. This method ensures that nodes are connected such that no node is inside the circumcircle of any triangle, thus forming a triangular mesh. Networks with randomly dispersed sensor nodes greatly benefit from this approach as it guarantees that the final network topology minimizes node overlaps and communication delays. Additionally, Delaunay triangulation is crucial for identifying coverage holes and network connectivity issues. By analyzing the connectivity of active nodes, regions with poor connectivity or coverage imbalances can be detected. In particular, large triangles in the mesh may indicate areas where coverage holes exist or where nodes may have exhausted their energy. These gaps can significantly impact the overall performance and coverage of the network. Based on the Delaunay triangulation, the total coverage quality of the network is evaluated by summing the coverage costs across all areas and subtracting the areas of triangles with detected coverage holes, and is numerically presented as

$$D = \sum_{i=1}^N \sum_{j=1}^M C_{i,j} - \sum_{\Delta_{ijk} \in T} \text{Area}(\Delta_{ijk}) \cdot \mathcal{H}(\text{Hole Detected}). \quad (11)$$

$$C_{i,j} = \begin{cases} 1 & \text{if the area at } (i,j) \text{ Area not covered or hole,} \\ 0 & \text{if the area at } (i,j) \text{ is covered.} \end{cases} \quad (12)$$

where  $C_{i,j}$  represents the cost associated with coverage and is defined based on the area of each triangle. The triangle cost is calculated using the area of the triangle formed by nodes  $N_i$ ,  $N_j$ , and  $N_k$ , where the area of a triangle with these three nodes is given by

$$\text{Area}(\Delta_{ijk}) = \frac{1}{2} |x_i(y_j - y_k) + x_j(y_k - y_i) + x_k(y_i - y_j)|. \quad (13)$$

## 5. Experimental Results and Comparison

The algorithm's results are assessed in this section, along with localization and WSN coverage tests. The performance of the proposed enhancements is evaluated based on the results obtained from the benchmark functions. In addition, the WSN coverage tests utilize the network coverage model to assess the optimization performance. These tests provide valuable insights into the effectiveness of the proposed changes in improving network coverage and overall algorithm efficiency.

### 5.1. Benchmark Function Tests

#### 5.1.1. Unimodal Benchmark Functions

Unimodal functions were used to assess the proposed algorithm's quick convergence time since they have unique global optima, which makes them appropriate for this use case. In particular, we evaluated the algorithm's performance using benchmark functions including Quartic Noise, Sphere, Schwefel 2.22, and Schwefel 2.21. While the Rosenbrock function, often known as the banana function, is a typical test case in optimization studies,

the Sphere function, a basic unimodal function, is frequently used to illustrate convergence speed. Because of their difficult landscapes with several local minima, the Schwefel 2.21, 2.22, and 1.2 functions are well-known benchmarks in optimization. As shown in Table 2, these functions are incorporated into our analysis to offer an additional comprehension of the algorithm's performance.

**Table 2.** Unimodal benchmark functions.

Function	Mathematical Expression	Search Space	Max. Velocity	Threshold
Schwefel 2.21	$f_1(x) = \max_i \{ x_i , 1 \leq i \leq 30\}$	$[-100, 100]$	20	0.01
Sphere	$f_2(x) = \sum_{i=1}^d (x_i)^2$	$[-100, 100]$	20	0.01
Schwefel 2.22	$f_3(x) = \sum_{i=1}^d  x_i  + \prod_{i=1}^d  x_i $	$[-10, 10]$	2	0.01
Schwefel 1.2	$f_4(x) = \sum_{i=1}^d \left( \sum_{j=1}^i x_j \right)$	$[-100, 100]$	20	0.01
Quartic Noise	$f_5(x) = \sum_{i=1}^d (ix_i^4 + \text{rand}[0, 1))$	$[-1.28, 1.28]$	0.25	0.1
Rosenbrock	$f_6(x) = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30, 30]$	6	0.1

### 5.1.2. Multimodal Benchmark Functions

Multimodal functions, on the other hand, are selected to evaluate the algorithm's capacity to prevent premature convergence since they make finding the global optimum more difficult. Applying VASF-PSO to such functions allowed for the evaluation of its convergence speed and solution consistency. For example, the interaction of its dimensional components makes the Griewank function challenging to optimize. Furthermore, the conventional multimodal penalized functions 1 and 2 make it more difficult to find the global optimum because of their misleading local optima that mimic the global minimum. A non-convex mathematical function, the Rastrigin function has a single global minimum and several local minima, making it a popular benchmark. The Ackley function is another complicated, non-convex function with several local minima and a dramatic peak indicating the global minimum. It is frequently used as a benchmark for algorithms that tackle challenging multimodal optimization issues. Xin-She Yang and Salomon are two more difficult multimodal benchmark functions. According to [58], the search space range for these test functions is carefully chosen to provide a consistent assessment, and the algorithm's halting conditions are based on a parameter unique to the issue called the threshold. For every benchmark function, the swarm size is 36 and the number of iterations is 1000, where the maximum dimension of the problem space, maximum  $d$ , is 30 for unimodal and multi-modal benchmark functions and 10 for CEC 2021 benchmark functions. These values are the same for all the variants used for comparison. The Wilcoxon and Friedman tests are used in performance analysis for fully evaluating the optimal outcomes of all algorithms on benchmark functions [59,60]. Table 3 provides the specifics of these multimodal benchmark functions, where the search range represents an area of the search space.

**Table 3.** Multimodal benchmark functions.

Function	Mathematical Expression	Search Space	Max. Velocity	Threshold
Penalized 2	$f_7(x) = \frac{1}{10} \left[ \sin^2(3\pi x_1) + \sum_{i=1}^d (x_i - 1)^2 (1 + \sin^2(3\pi x_i + 1)) \right. \\ \left. + (x_d - 1)^2 (1 + \sin(2\pi x_d)) + \sum_{i=1}^d u(x_i, 5, 100, 4) \right]$	$[-50, 50]$	10	0.01
Penalized 1	$f_8(x) = \frac{\pi}{n} \left[ 10 \sin(\pi y_1) + \sum_{i=1}^{d-1} (y_i - 1)^2 (1 + 10 \sin^2(\pi y_i + 1)) \right. \\ \left. + (y_d - 1)^2 + \sum_{i=1}^d u(x_i, 10, 100, 4) \right],$ $y_i = 1 + \frac{x_i + 1}{4}$	$[-50, 50]$	10	0.01
Griewank	$f_9(x) = \sum_{i=1}^d \frac{x_i^2}{4000} - \prod_{i=1}^d \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]$	120	0.1

Table 3. Cont.

Function	Mathematical Expression	Search Space	Max. Velocity	Threshold
Rastrigin	$f_{10}(x) = 10n + \sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i))$	$[-5.12, 5.12]$	1	0.1
Ackley	$f_{11}(x) = -ae^{-b\sqrt{\frac{1}{d}\sum_{i=1}^d x_i^2}} - e^{\frac{1}{d}\sum \cos(cx_i)} + a + e$	$[-32, 32]$	6.4	0.01
Salomon	$f_{12}(x) = 1 - \cos(2\pi\sqrt{\sum_{i=1}^n x_i^2}) + 0.1\sqrt{\sum_{i=1}^n x_i^2}$	$[-20, 20]$	4	0.1
Xin-She Yang	$f_{13}(x) = \sum  x_i e^{-\sum \sin(x_i^2)}$	$[-5, 5]$	1	0.1

### 5.1.3. CEC 2021 Benchmark Functions

The algorithms were evaluated with a set of functions that are frequently seen in recent CEC 2021 test suites. For example, the Shifted and fully Rotated Levy function, the Shifted and fully Rotated Expanded Schaffer's f6 function, and the Shifted and whole Rotated Zakharov function. The algorithm was also tested on various optimization problems using the Hybrid function 2 (N = 6) and the Composition function 4 (N = 6). These test functions thoroughly evaluate the algorithm's capabilities, spanning various complexity levels and corresponding with the kinds of issues present in the CEC test suites [61]. Table 4 provides the specifics of these CEC 2021 benchmark functions.

Table 4. Functions and parameters from the CEC 2021 benchmark set.

No	Function Name	Optimal Value
$f_{14}(x)$	Shifted and full Rotated Zakharov Function	300
$f_{15}(x)$	Shifted and full Rotated Expanded Schaffer's f6	600
$f_{16}(x)$	Shifted and full Rotated Levy Function	900
$f_{17}(x)$	Hybrid Function 2 (N = 6)	2000
$f_{18}(x)$	Composition Function 4 (N = 6)	2700

### 5.2. Outcome of Related PSO Algorithms

Tables 5–7 present the results of a comparison between several variants of PSO across multiple benchmark functions. Six algorithms, traditional PSO [62], linear decreasing inertia weight (LDIW) [63], time-varying acceleration coefficient (TVAC) [57], adaptive weighted delay velocity (AWDV) [64], adaptive weighted PSO (AWPSO) [65], and VASF-PSO, are used to evaluate each function with different dimensions. Table 8 presents the results of a comparison between several variants of PSO across CEC 2021 benchmark functions. For every algorithm, the table presents two performance metrics: Mean, which represents the average solution quality, and standard deviation (SD), which indicates the variation in outcomes. A lower Mean value signifies better optimization performance, while a lower SD indicates greater stability. The findings demonstrate that VASF-PSO consistently achieves the lowest values across most functions, exhibiting superior convergence speed and robustness compared to other methods.

Table 5. Comparison of algorithms across different functions (D = 10).

Function	PSO [66]		LDIW [67]		TVAC [68]		AWDV [64]		AWPSO [65]		VASF-PSO	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
$f_1(x)$	$8.61 \times 10^{-30}$	$4.98 \times 10^{-29}$	$2.07 \times 10^{-31}$	$1.69 \times 10^{-26}$	$6.98 \times 10^{-53}$	$8.56 \times 10^{-45}$	$2.76 \times 10^{-66}$	$6.23 \times 10^{-57}$	$2.68 \times 10^{-72}$	$5.34 \times 10^{-63}$	$6.2 \times 10^{-95}$	$6.1 \times 10^{-94}$
$f_2(x)$	$4.83 \times 10^{-26}$	$5.83 \times 10^{-19}$	$3.2 \times 10^{-32}$	$6.23 \times 10^{-24}$	$3.97 \times 10^{-20}$	$5.17 \times 10^{-16}$	$7.50 \times 10^{-22}$	$7.70 \times 10^{-19}$	$2.89 \times 10^{-28}$	$4.96 \times 10^{-21}$	$2.61 \times 10^{-41}$	$2.58 \times 10^{-40}$
$f_3(x)$	$6.56 \times 10^{-21}$	$7.66 \times 10^{-15}$	$2.94 \times 10^{-15}$	$2.94 \times 10^{-9}$	$5.29 \times 10^{-20}$	$5.29 \times 10^{-11}$	$1.18 \times 10^{-38}$	$1.18 \times 10^{-29}$	$1.32 \times 10^{-18}$	$1.32 \times 10^{-13}$	$8.34 \times 10^{-39}$	$8.64 \times 10^{-38}$
$f_4(x)$	$7.43 \times 10^{-31}$	$7.43 \times 10^{-28}$	$6.21 \times 10^{-24}$	$6.21 \times 10^{-18}$	$5.12 \times 10^{-19}$	$5.12 \times 10^{-12}$	$4.06 \times 10^{-26}$	$4.06 \times 10^{-20}$	$1.78 \times 10^{-29}$	$1.78 \times 10^{-25}$	$1.27 \times 10^{-42}$	$3.11 \times 10^{-42}$
$f_5(x)$	$7.05 \times 10^{-2}$	1.67	$7.67 \times 10^{-1}$	5.04	$4.78 \times 10^0$	3.10	$3.76 \times 10^{-2}$	1.11	$2.39 \times 10^{-3}$	1.34	$1.23 \times 10^{-4}$	$2.43 \times 10^{-4}$
$f_6(x)$	$1.7 \times 10^{-2}$	7.05	$7.0 \times 10^{-3}$	2.31	$7.6 \times 10^{-3}$	2.56	$7.6 \times 10^{-3}$	2.78	$8.3 \times 10^{-3}$	$9.45 \times 10^{-1}$	$1.04 \times 10^{-3}$	$4.89 \times 10^{-3}$
$f_7(x)$	$1.36 \times 10^{-32}$	$1.39 \times 10^{-27}$	$1.39 \times 10^{-32}$	$3.56 \times 10^{-28}$	$1.34 \times 10^{-32}$	$4.78 \times 10^{-29}$	$1.34 \times 10^{-32}$	$1.56 \times 10^{-29}$	$1.34 \times 10^{-32}$	$4.44 \times 10^{-30}$	$1.31 \times 10^{-32}$	$2.45 \times 10^{-32}$
$f_8(x)$	$2.36 \times 10^{-31}$	$3.85 \times 10^{-14}$	$3.85 \times 10^{-19}$	$6.45 \times 10^{-11}$	$2.36 \times 10^{-31}$	$5.56 \times 10^{-26}$	$2.36 \times 10^{-31}$	$4.16 \times 10^{-28}$	$2.30 \times 10^{-31}$	$3.40 \times 10^{-29}$	$3.11 \times 10^{-31}$	$5.12 \times 10^{-31}$
$f_9(x)$	$1.42 \times 10^{-2}$	1.50	$1.50 \times 10^{-2}$	2.56	$7.51 \times 10^{-3}$	1.59	$4.32 \times 10^{-3}$	$3.45 \times 10^{-2}$	$3.54 \times 10^{-3}$	$8.45 \times 10^{-2}$	$1.17 \times 10^{-3}$	$3.39 \times 10^{-3}$
$f_{10}(x)$	$1.09 \times 10^{-4}$	4.86	$4.86 \times 10^{-2}$	4.56	$8.21 \times 10^{-3}$	9.45	$5.74 \times 10^{-3}$	$8.47 \times 10^{-1}$	$7.34 \times 10^{-3}$	$8.45 \times 10^{-1}$	$1.09 \times 10^{-9}$	$2.67 \times 10^{-8}$
$f_{11}(x)$	$5.38 \times 10^{-5}$	7.88	$6.81 \times 10^{-4}$	$4.77 \times 10^{-1}$	$7.66 \times 10^{-3}$	$8.34 \times 10^{-1}$	$3.35 \times 10^{-7}$	$5.57 \times 10^{-2}$	$2.41 \times 10^{-8}$	$7.61 \times 10^{-3}$	$3.17 \times 10^{-11}$	$4.67 \times 10^{-10}$
$f_{12}(x)$	$3.11 \times 10^{-1}$	6.34	$5.12 \times 10^{-1}$	8.25	$6.98 \times 10^{-1}$	7.88	$3.08 \times 10^{-1}$	5.98	$2.16 \times 10^{-1}$	4.34	$8.08 \times 10^{-2}$	$9.98 \times 10^{-2}$
$f_{13}(x)$	$8.56 \times 10^{-1}$	9.56	$7.68 \times 10^{-1}$	7.78	$3.61 \times 10^{-2}$	5.34	$2.81 \times 10^{-2}$	4.78	$3.2 \times 10^{-2}$	$9.22 \times 10^{-1}$	$4.23 \times 10^{-3}$	$5.45 \times 10^{-3}$



**Table 6.** Comparison of algorithms across different functions (D = 20).

Function	PSO [66]		LDIW [67]		TVAC [68]		AWDV [64]		AWPSO [65]		VASF-PSO	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
$f_1(x)$	$2.85 \times 10^{-25}$	$3.23 \times 10^{-24}$	$5.31 \times 10^{-29}$	$4.14 \times 10^{-23}$	$2.98 \times 10^{-51}$	$7.34 \times 10^{-43}$	$3.08 \times 10^{-64}$	$7.25 \times 10^{-55}$	$2.34 \times 10^{-70}$	$3.55 \times 10^{-60}$	$4.95 \times 10^{-91}$	$5.12 \times 10^{-90}$
$f_2(x)$	$2.03 \times 10^{-21}$	$4.89 \times 10^{-18}$	$2.79 \times 10^{-30}$	$5.11 \times 10^{-21}$	$1.91 \times 10^{-18}$	$3.15 \times 10^{-15}$	$6.56 \times 10^{-20}$	$6.12 \times 10^{-16}$	$1.79 \times 10^{-27}$	$3.12 \times 10^{-18}$	$1.23 \times 10^{-39}$	$1.21 \times 10^{-38}$
$f_3(x)$	$4.92 \times 10^{-17}$	$6.28 \times 10^{-13}$	$1.87 \times 10^{-13}$	$1.54 \times 10^{-7}$	$4.39 \times 10^{-18}$	$3.67 \times 10^{-9}$	$8.01 \times 10^{-36}$	$8.12 \times 10^{-28}$	$1.12 \times 10^{-16}$	$1.42 \times 10^{-11}$	$5.12 \times 10^{-36}$	$5.32 \times 10^{-35}$
$f_4(x)$	$6.31 \times 10^{-28}$	$6.14 \times 10^{-25}$	$3.89 \times 10^{-21}$	$3.68 \times 10^{-16}$	$4.22 \times 10^{-18}$	$3.44 \times 10^{-11}$	$1.29 \times 10^{-24}$	$1.01 \times 10^{-19}$	$1.54 \times 10^{-28}$	$1.44 \times 10^{-23}$	$1.13 \times 10^{-41}$	$2.88 \times 10^{-41}$
$f_5(x)$	$3.67 \times 10^{-1}$	0.85	$5.49 \times 10^{-1}$	3.45	$3.21 \times 10^{-1}$	2.09	$2.59 \times 10^{-2}$	0.89	$1.15 \times 10^{-3}$	0.77	$8.39 \times 10^{-5}$	$1.43 \times 10^{-4}$
$f_6(x)$	$1.03 \times 10^{-2}$	3.25	$4.21 \times 10^{-3}$	1.87	$3.11 \times 10^{-3}$	1.98	$4.24 \times 10^{-3}$	2.44	$5.18 \times 10^{-3}$	0.65	$9.21 \times 10^{-4}$	$3.56 \times 10^{-3}$
$f_7(x)$	$9.23 \times 10^{-31}$	$1.02 \times 10^{-26}$	$1.04 \times 10^{-31}$	$2.58 \times 10^{-27}$	$1.13 \times 10^{-31}$	$3.21 \times 10^{-28}$	$1.22 \times 10^{-32}$	$1.34 \times 10^{-30}$	$1.06 \times 10^{-31}$	$2.03 \times 10^{-29}$	$1.09 \times 10^{-31}$	$1.51 \times 10^{-31}$
$f_8(x)$	$1.94 \times 10^{-30}$	$3.14 \times 10^{-13}$	$2.99 \times 10^{-18}$	$5.11 \times 10^{-10}$	$2.14 \times 10^{-30}$	$4.55 \times 10^{-24}$	$2.12 \times 10^{-30}$	$3.12 \times 10^{-26}$	$2.18 \times 10^{-30}$	$2.87 \times 10^{-27}$	$3.01 \times 10^{-30}$	$4.79 \times 10^{-30}$
$f_9(x)$	$1.23 \times 10^{-2}$	1.07	$1.08 \times 10^{-2}$	2.23	$4.89 \times 10^{-3}$	1.26	$3.21 \times 10^{-3}$	$3.12 \times 10^{-2}$	$2.13 \times 10^{-3}$	$5.21 \times 10^{-2}$	$8.94 \times 10^{-4}$	$2.89 \times 10^{-3}$
$f_{10}(x)$	$9.21 \times 10^{-5}$	4.34	$4.13 \times 10^{-2}$	3.94	$5.47 \times 10^{-3}$	8.12	$4.57 \times 10^{-3}$	0.59	$6.88 \times 10^{-3}$	0.68	$7.92 \times 10^{-9}$	$1.45 \times 10^{-7}$
$f_{11}(x)$	$3.88 \times 10^{-4}$	6.23	$5.93 \times 10^{-3}$	3.01	$7.01 \times 10^{-2}$	6.23	$3.01 \times 10^{-6}$	0.05	$2.58 \times 10^{-7}$	$5.91 \times 10^{-3}$	$3.21 \times 10^{-10}$	$4.13 \times 10^{-9}$
$f_{12}(x)$	$2.31 \times 10^{-1}$	4.54	$4.34 \times 10^{-1}$	7.32	$5.87 \times 10^{-1}$	6.89	$2.81 \times 10^{-1}$	5.21	$1.68 \times 10^{-1}$	2.32	$5.11 \times 10^{-2}$	$7.11 \times 10^{-2}$
$f_{13}(x)$	$6.93 \times 10^{-1}$	8.54	$6.23 \times 10^{-1}$	6.44	$1.22 \times 10^{-2}$	3.56	$2.32 \times 10^{-2}$	3.01	$1.23 \times 10^{-2}$	0.80	$3.79 \times 10^{-2}$	$4.01 \times 10^{-2}$

**Table 7.** Comparison of algorithms across different functions (D = 30).

Function	PSO [66]		LDIW [67]		TVAC [68]		AWDV [64]		AWPSO [65]		VASF-PSO	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
$f_1(x)$	$5.93 \times 10^{-25}$	$6.45 \times 10^{-24}$	$1.34 \times 10^{-28}$	$9.74 \times 10^{-22}$	$4.57 \times 10^{-51}$	$6.12 \times 10^{-43}$	$4.78 \times 10^{-62}$	$8.03 \times 10^{-54}$	$3.21 \times 10^{-68}$	$5.47 \times 10^{-57}$	$6.22 \times 10^{-90}$	$6.58 \times 10^{-89}$
$f_2(x)$	$1.93 \times 10^{-20}$	$3.45 \times 10^{-17}$	$1.65 \times 10^{-29}$	$3.45 \times 10^{-21}$	$1.78 \times 10^{-17}$	$2.98 \times 10^{-14}$	$5.23 \times 10^{-19}$	$5.89 \times 10^{-15}$	$1.22 \times 10^{-27}$	$3.21 \times 10^{-17}$	$1.16 \times 10^{-39}$	$1.14 \times 10^{-38}$
$f_3(x)$	$3.27 \times 10^{-16}$	$4.12 \times 10^{-12}$	$1.67 \times 10^{-12}$	$1.12 \times 10^{-6}$	$3.12 \times 10^{-17}$	$2.34 \times 10^{-8}$	$7.68 \times 10^{-34}$	$7.43 \times 10^{-26}$	$1.14 \times 10^{-15}$	$1.21 \times 10^{-10}$	$5.61 \times 10^{-34}$	$5.85 \times 10^{-33}$
$f_4(x)$	$4.72 \times 10^{-27}$	$5.81 \times 10^{-24}$	$2.58 \times 10^{-19}$	$2.34 \times 10^{-14}$	$3.14 \times 10^{-17}$	$2.56 \times 10^{-10}$	$1.06 \times 10^{-23}$	$8.97 \times 10^{-18}$	$1.23 \times 10^{-27}$	$1.12 \times 10^{-22}$	$1.09 \times 10^{-40}$	$2.76 \times 10^{-40}$
$f_5(x)$	$4.09 \times 10^{-1}$	1.22	$4.88 \times 10^{-1}$	3.67	$4.22 \times 10^{-1}$	2.65	$3.45 \times 10^{-2}$	0.72	$1.32 \times 10^{-3}$	0.62	$1.15 \times 10^{-4}$	$2.56 \times 10^{-4}$
$f_6(x)$	$1.31 \times 10^{-2}$	2.99	$5.23 \times 10^{-3}$	1.43	$4.09 \times 10^{-3}$	1.97	$5.18 \times 10^{-3}$	2.23	$6.11 \times 10^{-3}$	$5.34 \times 10^{-1}$	$1.32 \times 10^{-3}$	$4.56 \times 10^{-3}$
$f_7(x)$	$1.34 \times 10^{-30}$	$1.29 \times 10^{-26}$	$1.02 \times 10^{-32}$	$2.43 \times 10^{-28}$	$1.21 \times 10^{-32}$	$3.76 \times 10^{-29}$	$1.15 \times 10^{-33}$	$1.67 \times 10^{-31}$	$1.28 \times 10^{-32}$	$2.89 \times 10^{-29}$	$1.12 \times 10^{-32}$	$1.56 \times 10^{-32}$
$f_8(x)$	$1.56 \times 10^{-29}$	$3.21 \times 10^{-12}$	$3.12 \times 10^{-18}$	$5.12 \times 10^{-9}$	$2.04 \times 10^{-29}$	$5.66 \times 10^{-23}$	$2.02 \times 10^{-29}$	$4.56 \times 10^{-25}$	$2.19 \times 10^{-29}$	$2.94 \times 10^{-26}$	$3.12 \times 10^{-29}$	$4.34 \times 10^{-29}$
$f_9(x)$	$1.98 \times 10^{-2}$	1.56	$2.56 \times 10^{-2}$	3.12	$6.72 \times 10^{-3}$	1.68	$3.89 \times 10^{-3}$	$3.56 \times 10^{-2}$	$3.01 \times 10^{-3}$	$7.61 \times 10^{-2}$	$1.23 \times 10^{-3}$	$3.01 \times 10^{-3}$
$f_{10}(x)$	$8.76 \times 10^{-5}$	4.12	$3.68 \times 10^{-2}$	3.43	$4.67 \times 10^{-3}$	7.89	$4.25 \times 10^{-3}$	$6.12 \times 10^{-1}$	$5.97 \times 10^{-3}$	$6.01 \times 10^{-1}$	$8.76 \times 10^{-9}$	$1.23 \times 10^{-7}$
$f_{11}(x)$	$2.45 \times 10^{-4}$	5.76	$5.29 \times 10^{-3}$	2.89	$7.21 \times 10^{-2}$	5.67	$2.76 \times 10^{-6}$	$4.45 \times 10^{-2}$	$2.98 \times 10^{-7}$	$5.32 \times 10^{-3}$	$2.12 \times 10^{-10}$	$4.76 \times 10^{-9}$
$f_{12}(x)$	$1.56 \times 10^{-1}$	3.78	$3.12 \times 10^{-1}$	5.67	$4.67 \times 10^{-1}$	5.01	$2.19 \times 10^{-1}$	4.78	$1.12 \times 10^{-1}$	1.88	$4.56 \times 10^{-2}$	$6.56 \times 10^{-2}$
$f_{13}(x)$	$7.34 \times 10^{-1}$	8.56	$6.87 \times 10^{-1}$	6.21	$2.56 \times 10^{-2}$	3.12	$2.12 \times 10^{-2}$	2.45	$2.31 \times 10^{-2}$	$7.12 \times 10^{-1}$	$3.92 \times 10^{-2}$	$4.76 \times 10^{-2}$

**Table 8.** Selected CEC 2021 benchmark functions (D = 10).

Function	PSO [66]		LDIW [67]		TVAC [68]		AWDV [64]		AWPSO [65]		VASF-PSO	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
$f_{14}(x)$	$3.00 \times 10^2$	$4.44 \times 10^{-5}$	$7.71 \times 10^3$	$1.11 \times 10^3$	$6.66 \times 10^2$	$8.49 \times 10^2$	$5.45 \times 10^2$	$6.78 \times 10^3$	$5.04 \times 10^2$	$6.21 \times 10^3$	$3.20 \times 10^2$	$3.11 \times 10^2$
$f_{15}(x)$	$6.02 \times 10^2$	$3.35 \times 10^1$	$6.00 \times 10^2$	$7.8 \times 10^{-1}$	$6.27 \times 10^2$	$6.19 \times 10^2$	$6.10 \times 10^2$	$7.70 \times 10^2$	$5.80 \times 10^2$	$6.66 \times 10^2$	$1.88 \times 10^1$	$4.99 \times 10^0$
$f_{16}(x)$	$9.00 \times 10^2$	$2.73 \times 10^{-2}$	$9.01 \times 10^2$	$7.35 \times 10^{-1}$	$4.33 \times 10^3$	$9.68 \times 10^2$	$5.45 \times 10^2$	$6.45 \times 10^3$	$3.44 \times 10^2$	$6.45 \times 10^3$	$3.14 \times 10^2$	$3.74 \times 10^1$
$f_{17}(x)$	$2.02 \times 10^3$	$1.22 \times 10^2$	$2.20 \times 10^3$	$2.01 \times 10^{-2}$	$1.64 \times 10^3$	$2.23 \times 10^2$	$3.56 \times 10^3$	$4.34 \times 10^1$	$3.45 \times 10^3$	$2.64 \times 10^1$	$2.02 \times 10^3$	$8.86 \times 10^0$
$f_{18}(x)$	$2.86 \times 10^3$	$2.59 \times 10^1$	$2.86 \times 10^3$	$2.06 \times 10^{-1}$	$2.88 \times 10^3$	$2.20 \times 10^2$	$2.92 \times 10^3$	$2.15 \times 10^1$	$2.93 \times 10^3$	$2.08 \times 10^2$	$2.86 \times 10^3$	$1.13 \times 10^0$

The Friedman test yielded a  $p$ -value of  $1.7026 \times 10^{-6}$ , which is significantly lower than the commonly used significance level of 0.05 [69]. This strongly indicates that there is a noteworthy difference in the performance of the algorithms, leading us to reject the null hypothesis. As shown in Table 9, the proposed VASF-PSO outperforms all other algorithms, securing the top position with an overall rank of 1. It excels in both unimodal and multimodal functions, consistently achieving first place, thus demonstrating its superior performance compared to the other algorithms. In contrast, PSO performs the worst overall, ranking the lowest among all algorithms, while AWPSO takes second place. Graphical illustrations in Figure 4 complement Tables 5–7 by visually representing the comparative data. Heatmaps specifically highlight performance differences and metric distributions across various benchmark functions with different dimensions. Since the VASF-PSO algorithm is tested on minimization benchmark functions, lower values indicate better optimization performance.

**Table 9.** Friedman test results and algorithm rankings.

Algorithm	Unimodal Average Rank	Unimodal Overall Rank	Multimodal Average Rank	Multimodal Overall Rank	Overall Average Rank	Overall Rank
VASF-PSO	2.29	1	2.00	1	2.15	1
AWPSO	2.43	2	3.33	2	2.85	2
AWDV	2.86	3	3.00	4	2.92	3
LDIW	3.57	5	3.00	3	3.31	4
TVAC	3.86	4	3.83	6	3.85	5
PSO	6.00	6	5.83	5	5.92	6

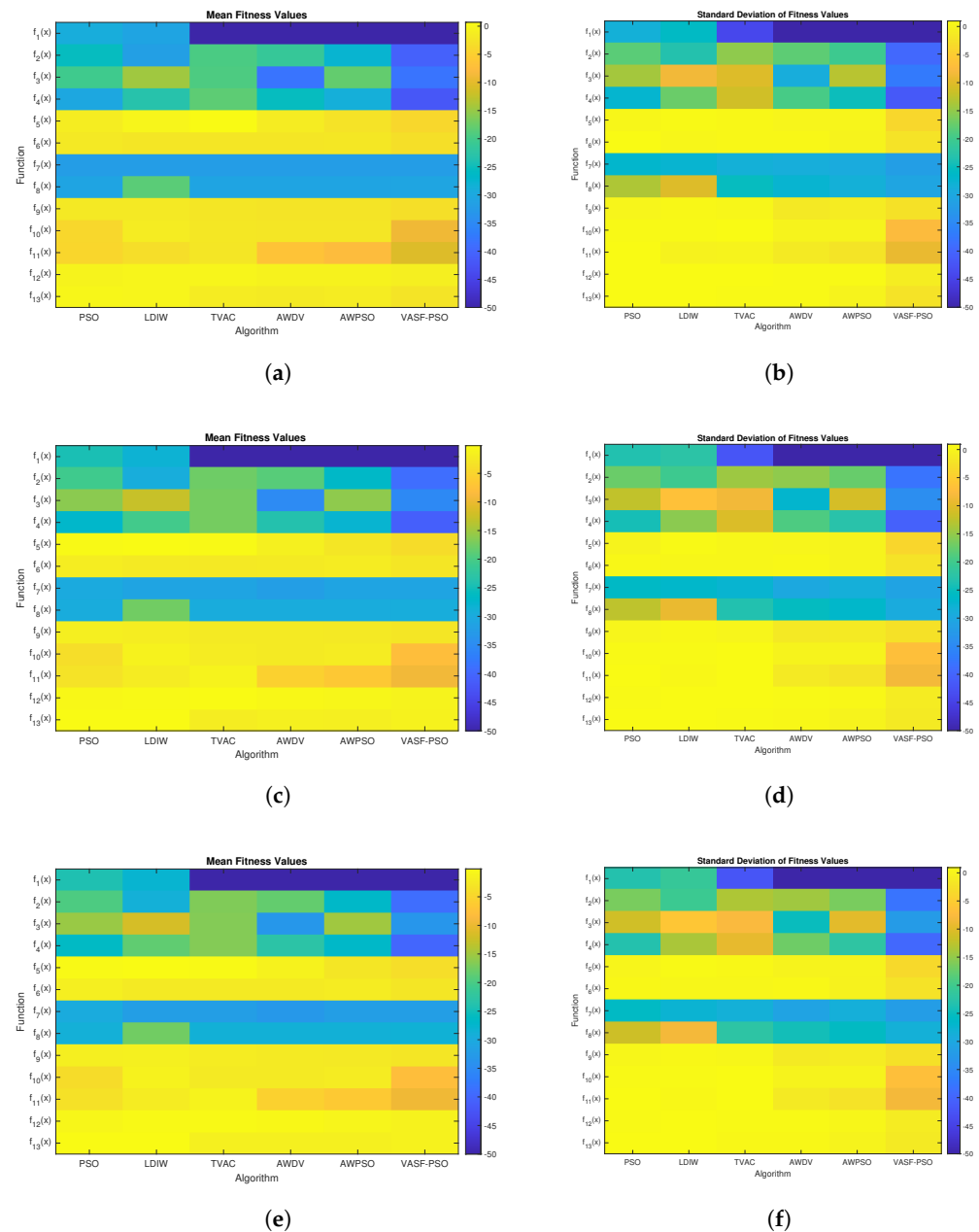
**Figure 4.** Graphical comparison of mean and standard deviation values corresponding to Tables 5–7. (a) Table 5 Mean values; (b) Table 5 SD values; (c) Table 6 Mean values; (d) Table 6 SD values; (e) Table 7 Mean values; (f) Table 7 SD values.

Table 10 shows the results of Wilcoxon Signed-Rank tests that compare the VASF-PSO algorithm's performance to that of five other algorithms: PSO, LDIW, TVAC, AWDV, and AWPSO. For each comparison, the  $p$ -value represents the probability of significance of the performance difference between VASF-PSO and the other methods. All  $p$ -values are less than 0.05, indicating that the results are significant. The  $p$ -value for comparing VASF-PSO

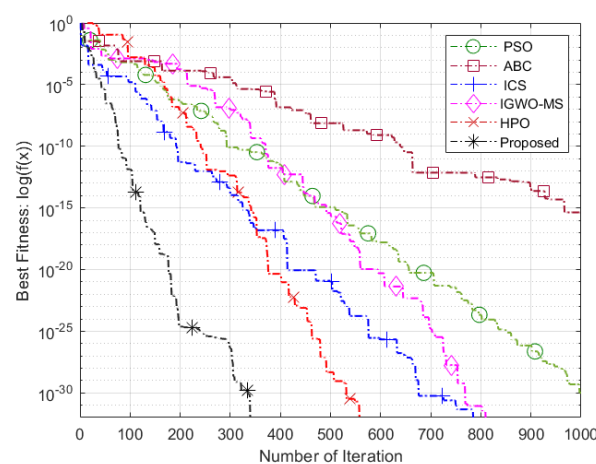
with PSO is 0.00073242; for LDIW, it is 0.00024414; for TVAC, it is 0.0012207; for AWDV, it is 0.001709; and for AWPSO, it is 0.0012207. For the CEC 2021 benchmark functions, the proposed VASF-PSO demonstrates competitive and often superior performance compared to other methods. While some comparisons did not yield statistically significant differences, VASF-PSO consistently matched or outperformed the alternatives. Notably, in several cases, especially when compared to the latest PSO variants such as AWDV and AWPSO, the  $p$ -values approached the significance threshold, proving that VASF-PSO exhibits enhanced effectiveness and robustness in challenging optimization scenarios.

**Table 10.** Wilcoxon test significant comparisons.

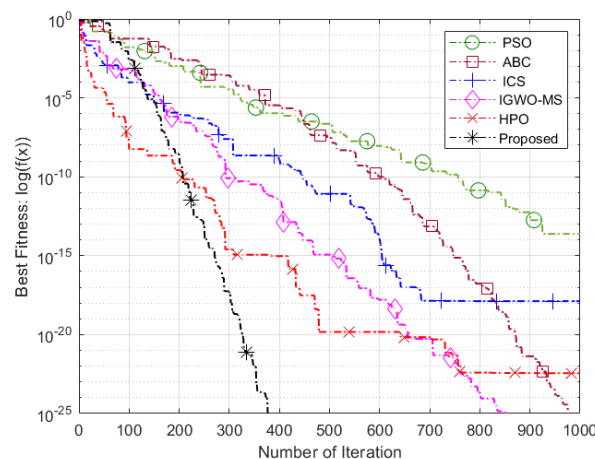
Comparison	Conclusion
VASF-PSO vs. PSO	VASF-PSO > PSO ( $p = 0.00073242$ )
VASF-PSO vs. LDIW	VASF-PSO > LDIW ( $p = 0.00024414$ )
VASF-PSO vs. TVAC	VASF-PSO > TVAC ( $p = 0.0012207$ )
VASF-PSO vs. AWDV	VASF-PSO > AWDV ( $p = 0.001709$ )
VASF-PSO vs. AWPSO	VASF-PSO > AWPSO ( $p = 0.0012207$ )

### 5.3. Evaluation of Intelligent Optimization Algorithms.

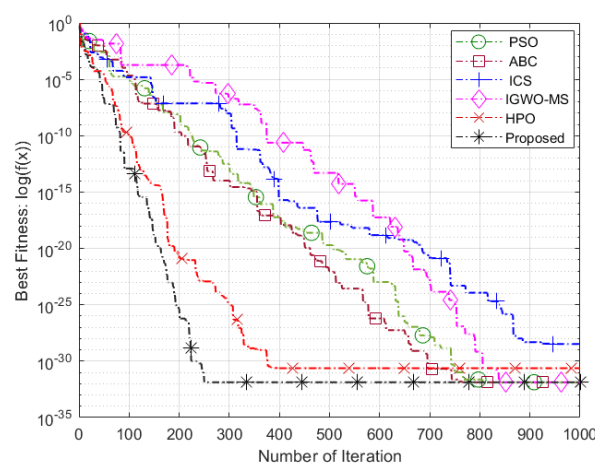
These benchmark functions' parameters pose significant challenges for algorithms during the algorithmic optimization procedure. They frequently result in fast convergence toward the nearest optimal value instead of the global optimum, for instance, trapping an algorithm in a local search stagnation situation. Consequently, these functions are perfect for evaluating an algorithm's ability to overcome local optima problems and keep seeking better solutions. The performance of the proposed algorithm compared to other optimization algorithms on benchmark functions is illustrated in Figures 5–7. The proposed algorithm demonstrates the fastest convergence, followed by the HPO algorithm, then IGWO-MS, ICS, ABC, and finally, the PSO algorithm. Using the Hammersley sequence for initialization is responsible for the proposed algorithm's higher performance. It is commonly known that this sequence produces equally distributed points throughout the search space, ensuring a well-balanced initial population. Because of its uniform distribution qualities, the algorithm can explore more effectively and converge more quickly. Additionally, the computational efficiency of the Hammersley sequence reduces overhead, enhancing the algorithm's performance and contributing to improved optimization outcomes.



**Figure 5.** Sphere function.



**Figure 6.** Schwefel 2.21 function.



**Figure 7.** Penalized function.

#### 5.4. WSN Coverage Performance Optimization

We compared the proposed VASF-PSO algorithm with traditional algorithms like PSO, ABC, ICS, IGWO-MS, and HPO to assess how well it performs in optimizing WSN node deployment while preserving a constant monitoring area and sensor types. The purpose of this comparison was to evaluate VASF-PSO's performance in enhancing wireless sensor network coverage and optimization.

The proposed algorithm is compared with traditional PSO, ABC, ICS, IGWO-MS, and HPO algorithms. The ABC algorithm, introduced by Karaboga in 2005, is a nature-inspired optimization method that mimics the foraging habits of honeybee colonies [70]. Similar to how bees locate high-quality nectar sources, sensor nodes cooperatively search for and exploit optimal deployment locations, making the ABC method well suited for WSN coverage optimization. Its simplicity and versatility are key advantages, enhancing coverage, energy efficiency, and sensor location optimization in WSNs. Additionally, an improved grey wolf optimizer with multi-strategies is employed to increase deployment efficiency and ensure effective coverage by leveraging the symmetry of sensor node sensing ranges [71]. The efficient implementation of the HPO algorithm in various WSN deployment scenarios is facilitated by its lightweight computational structure. Inspired by predator–prey interactions, this algorithm optimizes WSN coverage by dynamically repositioning sensor nodes (the hunters) to target and cover uncovered or optimal areas (the prey), thus maintaining a balanced exploration–exploitation approach [72]. This algorithm simulates the natural equilibrium where hunters actively seek prey by adjusting their locations relative

to the average prey population, focusing on those that are farther away. Furthermore, the improved cuckoo search (ICS) algorithm, inspired by the social behavior of animal groups, is combined with PSO, which updates solutions dimension by dimension, balances exploration and exploitation, and discretizes regional monitoring into point monitoring, ultimately improving coverage in WSNs with a limited number of sensors [73].

For case 1, the monitoring area is 2500 m<sup>2</sup>; for case 2, the monitoring area is 10,000 m<sup>2</sup>; and for case 3, the monitoring area is 22,500 m<sup>2</sup>. In case 1, 32 sensor nodes are used with a sensing range of 5 m; in case 2, 60 sensor nodes are used with a sensing range of 8 m; and in case 3, 36 sensor nodes are used with a sensing range of 15 m.

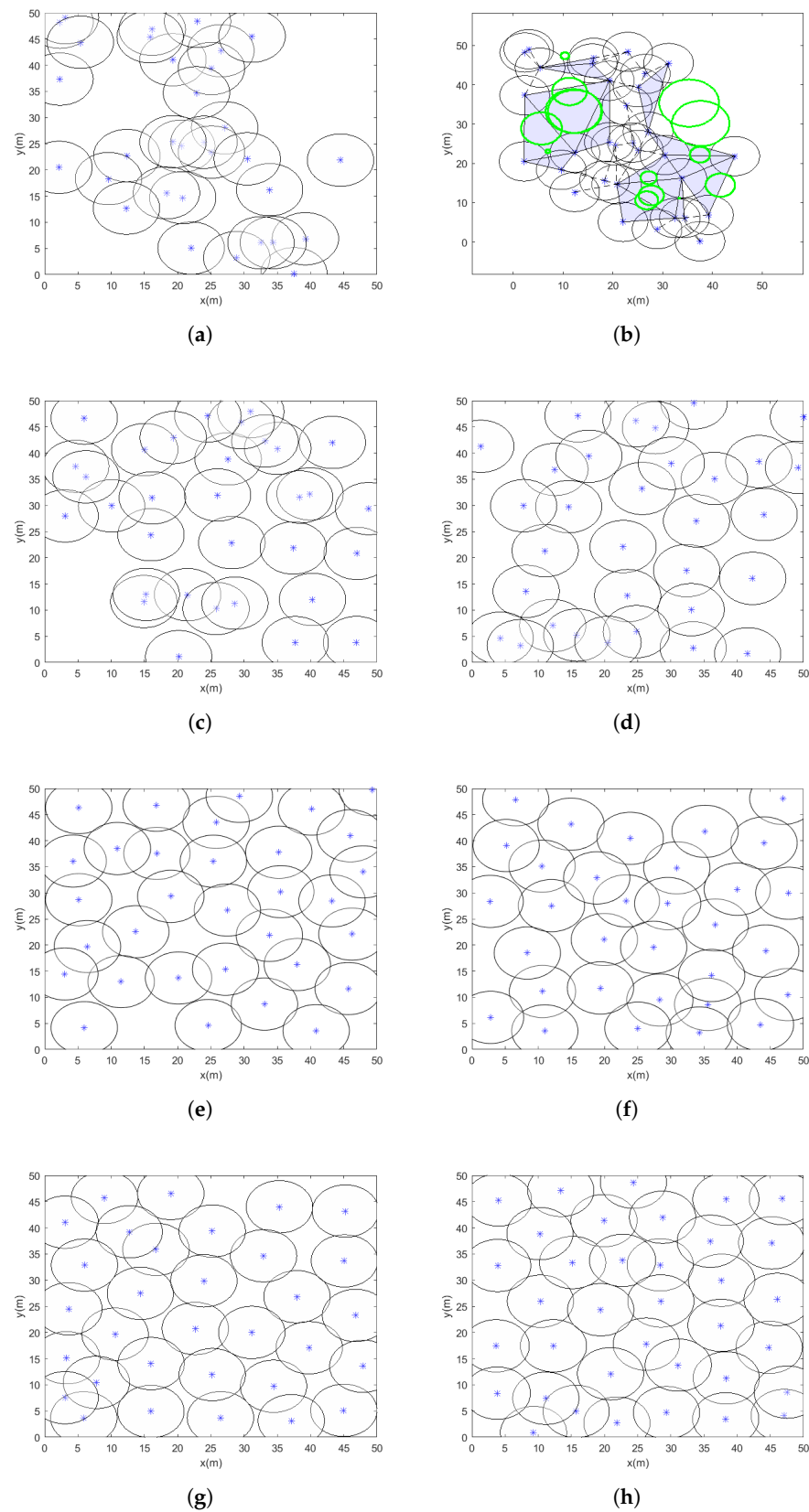
#### 5.4.1. Case 1

Table 11 presents the relevant optimization results for case 1, including the experimental conditions, which feature 32 sensor nodes with a coverage range of 5 m. The radius of the sensor nodes remains constant during the test, with the node placement for case 1 shown in Figure 8, the star is the center of node the black circle is it radius the blue triangle are its coverage gaps and the green circle are covergae hole. Over 20 independent experiments, the proposed VASF-PSO algorithm consistently achieves full coverage in the coverage area for case 1, demonstrating the algorithm's reliability in optimizing sensor placement. The VASF-PSO algorithm also shows rapid convergence and an efficient optimization process, as it covers the monitored region in fewer than 50 iterations on average.

In contrast, conventional optimization algorithms such as PSO, ABC, ICS, IGWO-MS, and HPO require more than 200 iterations to achieve full coverage, with coverage rates around 90% or lower. In comparison, the VASF-PSO algorithm consistently reaches a 99% coverage rate in fewer than 40 iterations. This significant reduction in the number of iterations highlights the improved efficiency and faster convergence of the VASF-PSO algorithm when compared to traditional algorithms.

**Table 11.** Simulation result analysis.

Parameters (m)			
	Case 1	Case 2	Case 3
Coverage Area	[50, 50]	[100, 100]	[150, 150]
Sensor Nodes	32	60	36
Sensing Radius	5 m	8 m	15 m
Coverage Holes	11	23	15
Coverage Rate			
Algorithm	Case 1	Case 2	Case 3
PSO	0.8856	0.8545	0.8278
ABC	0.8945	0.8434	0.8195
ICS	0.9105	0.8656	0.8345
IGWO-MS	0.9387	0.8813	0.8534
HPO	0.9456	0.9034	0.8967
Proposed	0.9945	0.9810	0.9690
Computation Time (s)			
Algorithm	Case 1	Case 2	Case 3
PSO	58.09	72.81	81.56
ABC	57.78	73.18	85.23
ICS	56.11	71.45	82.67
IGWO-MS	53.45	66.25	72.78
HPO	49.89	61.10	68.19
Proposed	38.98	43.56	48.88



**Figure 8.** Node deployment case 1: (a) the initial deployment of WSNs; (b) coverage hole in initial positions of nodes; (c) optimized deployment of WSNs by PSO; (d) optimized deployment of WSNs by ABC; (e) optimized deployment of WSNs by ICS; (f) optimized deployment of WSNs by IGWO-MS; (g) optimized deployment of WSNs by HPO; (h) optimized deployment of WSNs by the proposed algorithm.



#### 5.4.2. Case 2

In case 2, 60 sensor nodes with an 8 m sensing radius are used in the experiment; the sensor node radius stays constant during the test. In Figure 9, the node arrangement for case 2 is presented. Twenty different tests show that the proposed VASF-PSO algorithm is reliable in maximizing sensor placement, reliably achieving full coverage in the monitored region. The VASF-PSO method also demonstrates quick convergence, completing the region in an average of less than 70 iterations. This efficiency demonstrates how well the method can prevent premature convergence while keeping solution space exploration. From the experiments, we can see the coverage rate, demonstrating the stability and resilience of the optimization strategy. Furthermore, it appears that the algorithm successfully balances node density, preventing redundant overlaps and minimizing coverage holes, based on the uniform distribution of nodes shown in the end deployments.

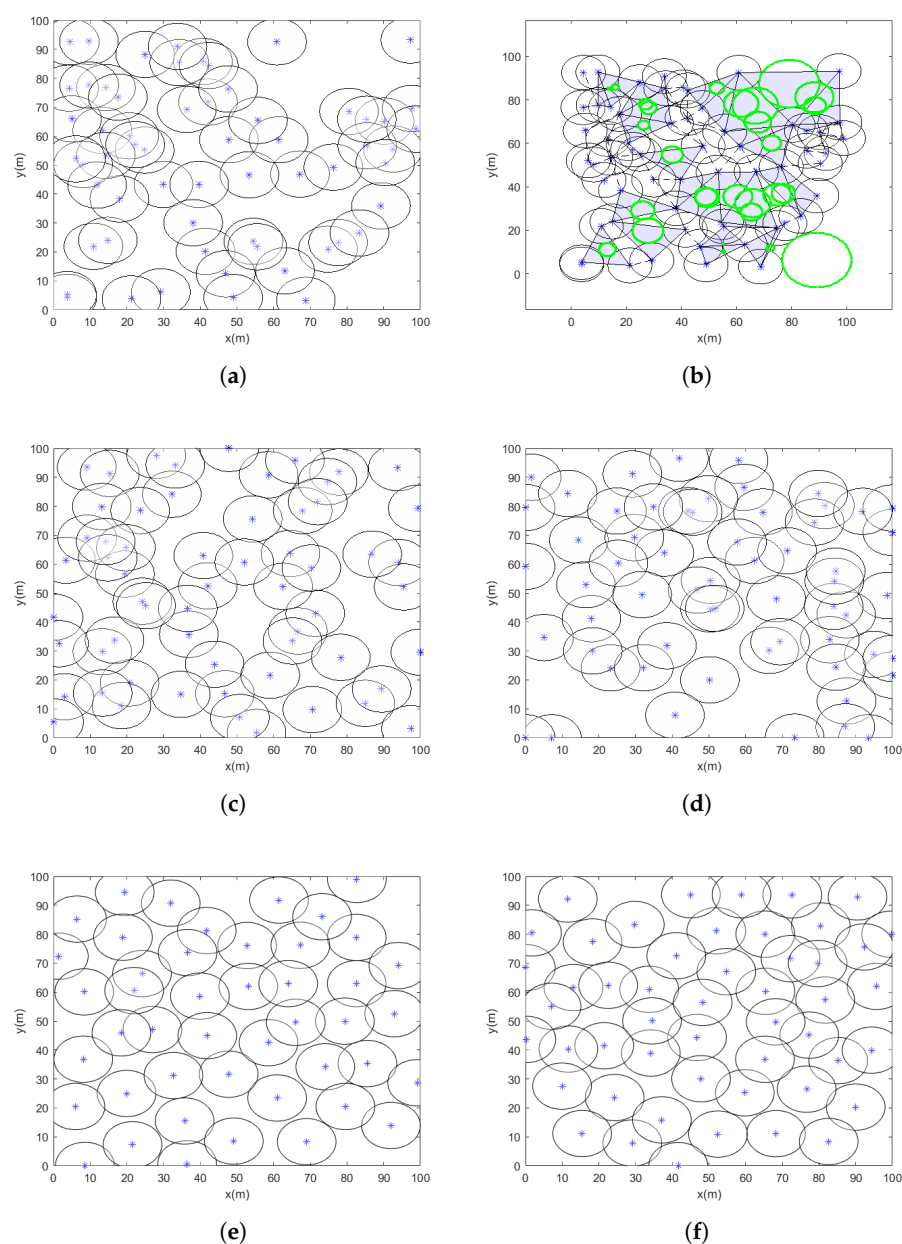
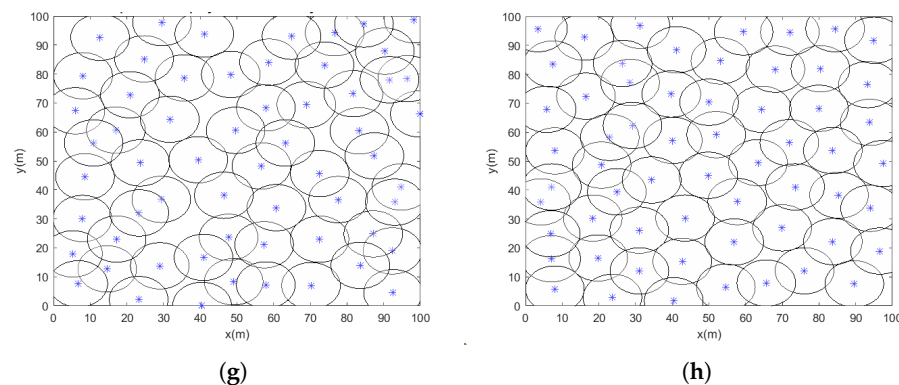


Figure 9. Cont.



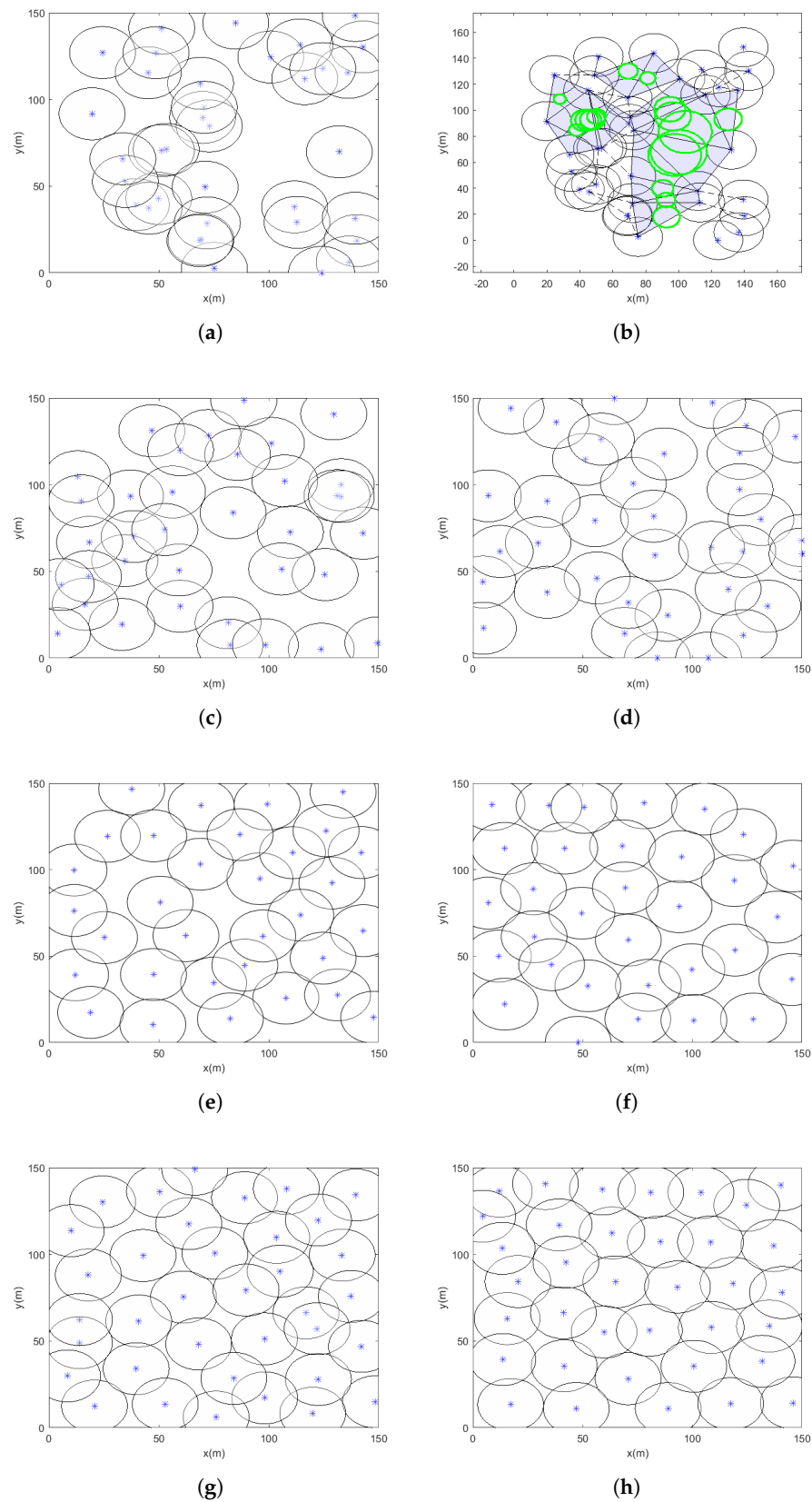
**Figure 9.** Node deployment case 2: (a) the initial deployment of WSNs; (b) coverage hole in initial positions of nodes; (c) optimized deployment of WSNs by PSO; (d) optimized deployment of WSNs by ABC; (e) optimized deployment of WSNs by ICS; (f) optimized deployment of WSNs by IGWO-MS; (g) optimized deployment of WSNs by HPO; (h) optimized deployment of WSNs by the proposed algorithm.

In comparison, conventional optimization algorithms such as PSO, ABC, ICS, IGWO-MS, and HPO require more than 290 iterations to attain full coverage, with coverage rates around 90% or lower. The VASF-PSO algorithm, on the other hand, consistently achieves 97% to 98% coverage in fewer than 60 iterations, highlighting its superior efficiency and coverage compared to traditional algorithms.

#### 5.4.3. Case 3

The experimental implementation in case 3 consists of 36 sensor nodes, each with a sensing radius of 15 m, which stays fixed for the entire duration of the experiment. Case 3's sensor node configuration is displayed in Figure 10. The proposed VASF-PSO algorithm continuously achieves full coverage of the monitored region during 20 separate test runs, demonstrating the algorithm's dependability in carefully positioning sensors for full coverage. The algorithm is capable of finding optimal solutions quickly since it exhibits efficient convergence, covering the full region in an average of less than 85 iterations.

The performance of VASF-PSO is significantly influenced by factors such as node count, node radius, and monitoring area size. Increasing the number of nodes improves coverage but also raises computational costs. Larger node radii result in better coverage, although the benefits begin to diminish beyond a certain point. Expanding the monitoring area typically reduces coverage rates and increases computation time. In contrast, traditional optimization methods like PSO, ABC, ICS, IGWO-MS, and HPO generally achieve coverage rates between 81% and 89% and require over 400 iterations to reach full coverage. In comparison, VASF-PSO consistently achieves a 96% coverage rate in fewer than 85 iterations, demonstrating its superior efficacy in achieving higher coverage with significantly fewer iterations than conventional methods. This illustrates its greater effectiveness and ability to obtain a greater coverage rate in a significantly smaller number of iterations, in contrast to traditional techniques. Table 11 displays the optimization results for cases 1, 2, and 3, which include the coverage area, coverage holes, coverage rate, and total computational time. Graphical illustrations provide additional insight into Figure 11 by visually summarizing simulation results across different coverage scenarios. The figures highlight key performance metrics such as coverage rate and computation time for various algorithms, clearly demonstrating the superior effectiveness and efficiency of the proposed VASF-PSO algorithm under varying area sizes and sensor node configurations.



**Figure 10.** Node deployment case 3: (a) the initial deployment of WSNs; (b) coverage hole in initial positions of nodes; (c) optimized deployment of WSNs by PSO; (d) optimized deployment of WSNs by ABC; (e) optimized deployment of WSNs by ICS; (f) optimized deployment of WSNs by IGWO-MS; (g) optimized deployment of WSNs by HPO; (h) optimized deployment of WSNs by the proposed algorithm.

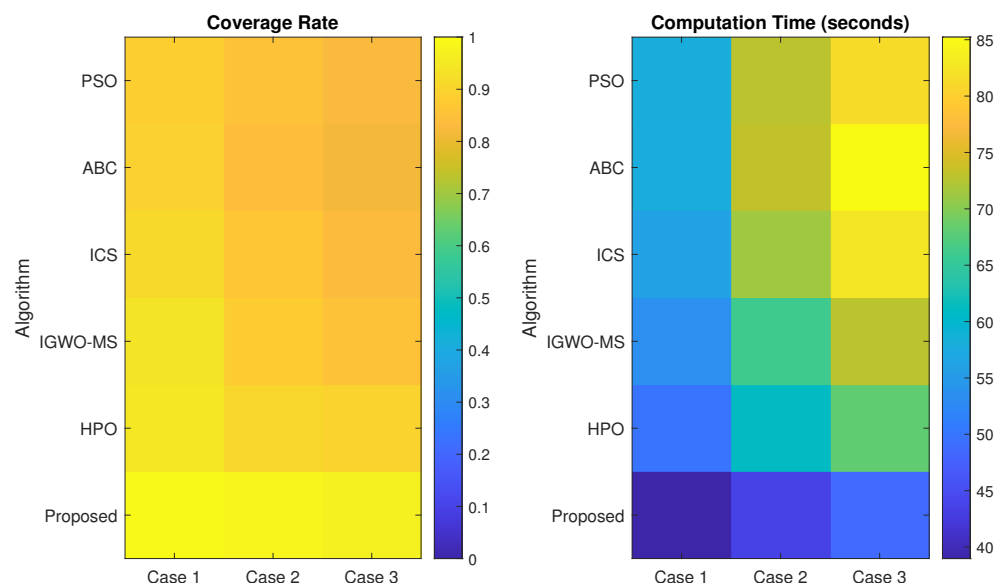


Figure 11. Result visualization of Table 11.

As shown in Figure 12, when the total number of nodes is small, the solution tends to have a high degree of redundant coverage, meaning that some areas are covered by multiple nodes. However, as the number of nodes increases, the PSO, ABC, ICS, IGWO-MS, and HPO algorithms still show noticeable blind spots in the deployment, indicating that certain regions remain uncovered or inadequately covered, even with the higher node density. In contrast, the VASF-PSO algorithm consistently achieves a higher coverage rate, effectively optimizing node placement and reducing blind spots more efficiently than the other four algorithms across various node density levels.

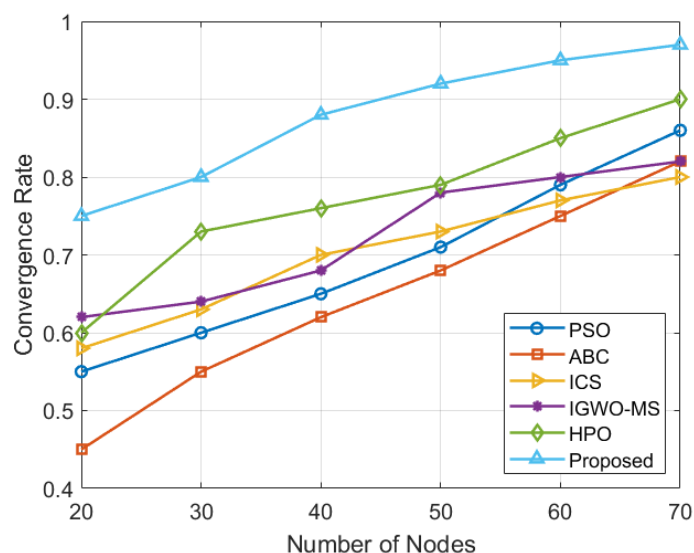


Figure 12. Node density and coverage area.

Comparing the VASF-PSO algorithm to more conventional optimization techniques like PSO, ABC, ICS, IGWO-MS, and HPO, the coverage rate for all three test scenarios is noticeably higher. In particular, case 1 shows a notable increase over traditional techniques, with the VASF-PSO algorithm achieving a 10% higher coverage rate. This increase becomes much more notable in case 2 when VASF-PSO's superior performance is demonstrated by a 14.71% greater coverage rate. In contrast to conventional algorithms, VASF-PSO further demonstrates its efficacy in enhancing coverage in case 3, where it achieves a 12.94%

increase in coverage rate. These results highlight the consistent and substantial advantages of the VASF-PSO algorithm in terms of coverage rate across a variety of scenarios.

## 6. Limitations and Future Work

The VASF-PSO algorithm demonstrates strong coverage performance across varying network scales but also exhibits important limitations. As the number of sensor nodes or the coverage area increases, the computational complexity of the optimization problem grows significantly, resulting in longer processing times and reduced scalability for extensive networks. This increased complexity can challenge real-time or resource-constrained applications. Furthermore, there is an inherent trade-off between maximizing coverage and minimizing energy consumption. Achieving high coverage in larger networks typically requires activating more nodes or increasing node utilization, which leads to higher overall energy expenditure and can significantly impact network lifetime and sustainability. Addressing these challenges through multi-objective optimization frameworks or adaptive energy-aware mechanisms could improve both scalability and energy efficiency. These represent promising directions for future work to enhance the practical applicability of the VASF-PSO algorithm.

## 7. Conclusions

This paper proposes the VASF-PSO algorithm for optimizing sensor node coverage in WSNs. Unlike traditional algorithms, which often struggle with local optima, VASF-PSO enhances coverage performance by improving search accuracy and minimizing the risk of getting stuck in local optima. Key features of the algorithm include Hammersley sequence-based initialization, a velocity-scaled adaptive search factor, and Delaunay triangulation for identifying and filling coverage gaps. Results from three test cases demonstrate the advantages of VASF-PSO. In case 1, with 32 nodes and a 5 m sensing radius, VASF-PSO achieved full coverage in fewer than 50 iterations, while previous algorithms required over 200 iterations. In case 2, with 60 nodes and an 8 m sensing radius, VASF-PSO consistently reached 97–98% coverage in under 60 iterations, compared to just 90% coverage after 290 iterations by conventional algorithms. In case 3, with 36 nodes and a 15 m sensing radius, VASF-PSO achieved 96% coverage in under 85 iterations, while traditional algorithms struggled to reach 89% coverage in over 400 iterations. These results demonstrate that VASF-PSO optimizes coverage more efficiently, requiring fewer iterations and offering improved coverage, outperforming traditional algorithms in WSN coverage optimization. Although VASF-PSO shows promising results, its performance depends on parameter tuning and assumes ideal conditions, without accounting for real-world uncertainties like sensor failures or communication constraints. Future work will focus on improving adaptability in dynamic environments, optimizing energy efficiency, and validating the algorithm through real-world implementations.

**Author Contributions:** Conceptualization: H.M. and H.N.; data curation: H.M.; formal analysis: H.M.; funding acquisition: H.N.; methodology: H.M. and H.N.; project administration: H.N.; resources: H.M.; software: H.M.; supervision: H.N.; validation: H.M. and H.N.; visualization: H.M.; writing—original draft: H.M.; writing—review and editing: H.M. and H.N. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2025-RS-2023-00258639) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation) and also in part by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (2022R1A2C1011862).

**Data Availability Statement:** Data are contained within this article.

**Conflicts of Interest:** The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

## Abbreviations

The following abbreviations are used in this manuscript:

PSO	Particle swarm optimization
WSNs	Wireless sensor networks
VASF-PSO	Velocity-scaled adaptive search factor PSO
HPO	Hunter–prey optimization
IGWO-MS	Improved grey wolf optimizer with multi-strategies
ICS	Improved cuckoo search
ABC	Artificial bee colony
QoS	Quality of service
BS	Base station
SNs	Sensor nodes

## References

- Oubbati, O.S.; Atiquzzaman, M.; Lim, H.; Rachedi, A.; Lakas, A. Synchronizing UAV teams for timely data collection and energy transfer by deep reinforcement learning. *IEEE Trans. Veh. Technol.* **2022**, *71*, 6682–6697. [\[CrossRef\]](#)
- Fascista, A. Toward integrated large-scale environmental monitoring using WSN/UAV/Crowdsensing: A review of applications, signal processing, and future perspectives. *Sensors* **2022**, *22*, 1824. [\[CrossRef\]](#) [\[PubMed\]](#)
- Bokhari, M.U.; Kareem, A.; Hanafi, B. Exploring Fault Tolerance Consensus for Wireless Sensor Networks: A Comprehensive Detailed Study. *J. Electr. Syst.* **2024**, *20*, 1653–1663. [\[CrossRef\]](#)
- Sheng, Z.; Mahapatra, C.; Zhu, C.; Leung, V.C. Recent advances in industrial wireless sensor networks toward efficient management in IoT. *IEEE Access* **2015**, *3*, 622–637. [\[CrossRef\]](#)
- Coboi, A.; Nguyen, M.T.; Primeiro, I.Z.; Van Nam, P.; Van Huy, B.; Ta, T.M.; Nguyen, T.T. Towards Multiple Sources for Energy Harvesting in Wireless Sensor Networks in Practical Applications. *Comput. Netw. Commun.* **2024**, *2*, 111–131. [\[CrossRef\]](#)
- Xia, Y.; Deng, X.; Yi, L.; Liu, S.; Yang, L.T.; Zhu, C.; Tang, X. A trust-based reliable confident information coverage model of wireless sensor networks for intelligent transportation. *IEEE Trans. Veh. Technol.* **2023**, *72*, 9542–9554. [\[CrossRef\]](#)
- Heath, R.W.; Kountouris, M.; Bai, T. Modeling heterogeneous network interference using Poisson point processes. *IEEE Trans. Signal Process.* **2013**, *61*, 4114–4126. [\[CrossRef\]](#)
- Shah, S.L.; Abbas, Z.H.; Abbas, G.; Muhammad, F.; Hussien, A.; Baker, T. An innovative clustering hierarchical protocol for data collection from remote wireless sensor networks based internet of things applications. *Sensors* **2023**, *23*, 5728. [\[CrossRef\]](#)
- Lima, M.M.; Sardinha, E.D.; Balico, L.N.; Oliveira, H.A. PATCH: Proactive approach to circumvent holes in wireless sensor networks. *Sensors* **2023**, *23*, 7862. [\[CrossRef\]](#)
- Gola, K.K.; Gupta, B. An energy-efficient quality of service (QOS) parameter-based void avoidance routing technique for underwater sensor networks. *Jordanian J. Comput. Inf. Technol.* **2019**, *5*. [\[CrossRef\]](#)
- Guleria, K.; Verma, A.K. Comprehensive review for energy efficient hierarchical routing protocols on wireless sensor networks. *Wirel. Netw.* **2019**, *25*, 1159–1183. [\[CrossRef\]](#)
- Ma, D.; Lan, G.; Hassan, M.; Hu, W.; Das, S.K. Sensing, computing, and communications for energy harvesting IoTs: A survey. *IEEE Commun. Surv. Tutor.* **2019**, *22*, 1222–1250. [\[CrossRef\]](#)
- Majid, M.; Habib, S.; Javed, A.R.; Rizwan, M.; Srivastava, G.; Gadekallu, T.R.; Lin, J.C.W. Applications of wireless sensor networks and internet of things frameworks in the industry revolution 4.0: A systematic literature review. *Sensors* **2022**, *22*, 2087. [\[CrossRef\]](#) [\[PubMed\]](#)
- Ray, P.P.; Mukherjee, M.; Shu, L. Internet of Things for Disaster Management: State-of-the-Art and Prospects. *IEEE Access* **2017**, *5*, 18818–18835. [\[CrossRef\]](#)
- Li, K.; Voicu, R.C.; Kanhere, S.S.; Ni, W.; Tovar, E. Energy Efficient Legitimate Wireless Surveillance of UAV Communications. *IEEE Trans. Veh. Technol.* **2019**, *68*, 2283–2293. [\[CrossRef\]](#)
- Kang, J.J.; Yang, W.; Dermody, G.; Ghasemian, M.; Adibi, S.; Haskell-Dowland, P. No Soldiers Left Behind: An IoT-Based Low-Power Military Mobile Health System Design. *IEEE Access* **2020**, *8*, 201498–201515. [\[CrossRef\]](#)



17. Osanaiye, O.A.; Alfa, A.S.; Hancke, G.P. Denial of Service Defence for Resource Availability in Wireless Sensor Networks. *IEEE Access* **2018**, *6*, 6975–7004. [\[CrossRef\]](#)
18. Raza, M.; Aslam, N.; Le-Minh, H.; Hussain, S.; Cao, Y.; Khan, N.M. A Critical Analysis of Research Potential, Challenges, and Future Directives in Industrial Wireless Sensor Networks. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 39–95. [\[CrossRef\]](#)
19. Jamshed, M.A.; Ali, K.; Abbasi, Q.H.; Imran, M.A.; Ur-Rehman, M. Challenges, applications, and future of wireless sensors in Internet of Things: A review. *IEEE Sens. J.* **2022**, *22*, 5482–5494. [\[CrossRef\]](#)
20. Al-Karaki, J.N.; Gawanmeh, A. The optimal deployment, coverage, and connectivity problems in wireless sensor networks: Revisited. *IEEE Access* **2017**, *5*, 18051–18065. [\[CrossRef\]](#)
21. Zhang, Z.; Mehmood, A.; Shu, L.; Huo, Z.; Zhang, Y.; Mukherjee, M. A survey on fault diagnosis in wireless sensor networks. *IEEE Access* **2018**, *6*, 11349–11364. [\[CrossRef\]](#)
22. Latif, K.; Javaid, N.; Ahmad, A.; Khan, Z.A.; Alrajeh, N.; Khan, M.I. On energy hole and coverage hole avoidance in underwater wireless sensor networks. *IEEE Sens. J.* **2016**, *16*, 4431–4442. [\[CrossRef\]](#)
23. Ekhlas, V.R.; Shirvani, M.H.; Dana, A.; Raeisi, N. Discrete grey wolf optimization algorithm for solving k-coverage problem in directional sensor networks with network lifetime maximization viewpoint. *Appl. Soft Comput.* **2023**, *146*, 110609. [\[CrossRef\]](#)
24. Shi, Y.; Yang, K.; Jiang, T.; Zhang, J.; Letaief, K.B. Communication-efficient edge AI: Algorithms and systems. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 2167–2191. [\[CrossRef\]](#)
25. Li, J.; Yang, L.; Wu, Q.; Lei, X.; Zhou, F.; Shu, F.; Mu, X.; Liu, Y.; Fan, P. Active RIS-aided NOMA-enabled space-air-ground integrated networks with cognitive radio. *IEEE J. Sel. Areas Commun.* **2024**, *43*, 314–333. [\[CrossRef\]](#)
26. Sharma, V.; Patel, R.; Bhadauria, H.S.; Prasad, D. NADS: Neighbor assisted deployment scheme for optimal placement of sensor nodes to achieve blanket coverage in wireless sensor network. *Wirel. Pers. Commun.* **2016**, *90*, 1903–1933. [\[CrossRef\]](#)
27. Pandey, D.; Kushwaha, V. An Exploratory Study of Optimization Techniques for Congestion Control in Wireless Sensor Networks. *Adhoc Sens. Wirel. Netw.* **2024**, *58*, 79.
28. Vu, H.N.; Pham, C.; Dung, N.M.; Ro, S. Detecting and tracking sinkholes using multi-level convolutional neural networks and data association. *IEEE Access* **2020**, *8*, 132625–132641. [\[CrossRef\]](#)
29. Guo, J.; Sun, Y.; Liu, T.; Li, Y.; Fei, T. An Optimization Coverage Strategy for Wireless Sensor Network Nodes Based on Path Loss and False Alarm Probability. *Sensors* **2025**, *25*, 396. [\[CrossRef\]](#)
30. Liu, Y.; Li, Q. Coverage algorithm based on perceived environment around nodes in mobile wireless sensor networks. *Wirel. Pers. Commun.* **2023**, *128*, 2725–2740. [\[CrossRef\]](#)
31. Neggaz, N.; Seyyedabbasi, A.; Hussien, A.G.; Rahim, M.; Beşkirli, M. Optimal nodes localization in wireless sensor networks using Nutcracker optimizer algorithms: Istanbul Area. *IEEE Access* **2024**, *12*, 67986–68002. [\[CrossRef\]](#)
32. Mishra, S.D.; Verma, D. Energy-efficient and reliable clustering with optimized scheduling and routing for wireless sensor networks. *Multimed. Tools Appl.* **2024**, *83*, 68107–68133. [\[CrossRef\]](#)
33. Yue, Y.G.; He, P. A comprehensive survey on the reliability of mobile wireless sensor networks: Taxonomy, challenges, and future directions. *Inf. Fusion* **2018**, *44*, 188–204. [\[CrossRef\]](#)
34. Ahmed, M.M.; Houssein, E.H.; Hassanien, A.E.; Taha, A.; Hassanien, E. Maximizing lifetime of wireless sensor networks based on whale optimization algorithm. In Proceedings of the International Conference on Advanced Intelligent Systems and Informatics, Cairo, Egypt, 9–11 September 2017; Springer: Berlin/Heidelberg, Germany, 2018; pp. 724–733.
35. Lakshmi, Y.V.; Singh, P.; Abouhawwash, M.; Mahajan, S.; Pandit, A.K.; Ahmed, A.B. Improved Chan algorithm based optimum UWB sensor node localization using hybrid particle swarm optimization. *IEEE Access* **2022**, *10*, 32546–32565. [\[CrossRef\]](#)
36. Wang, M.; Wang, X.; Jiang, K.; Fan, B. Reinforcement learning-enabled resampling particle swarm optimization for sensor relocation in reconfigurable WSNs. *IEEE Sens. J.* **2022**, *22*, 8257–8267. [\[CrossRef\]](#)
37. Zhang, Y.; Shen, W. A novel particle swarm optimization algorithm for k-coverage problems in wireless sensor networks. In Proceedings of the 2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Dalian, China, 5–7 May 2021; IEEE: New York, NY, USA, 2021; pp. 831–836.
38. Cao, L.; Yue, Y.; Cai, Y.; Zhang, Y. A novel coverage optimization strategy for heterogeneous wireless sensor networks based on connectivity and reliability. *IEEE Access* **2021**, *9*, 18424–18442. [\[CrossRef\]](#)
39. Yao, Y.; Liao, H.; Liu, M.; Yang, X. Coverage optimization strategy for 3-D wireless sensor networks based on improved sparrow search algorithm. *IEEE Sens. J.* **2023**, *23*, 23721–23733. [\[CrossRef\]](#)
40. Zhao, Q.; Li, C.; Zhu, D.; Xie, C. Coverage optimization of wireless sensor networks using combinations of PSO and chaos optimization. *Electronics* **2022**, *11*, 853. [\[CrossRef\]](#)
41. Gamal, M.; Mekky, N.E.; Soliman, H.; Hikal, N.A. Enhancing the lifetime of wireless sensor networks using fuzzy logic LEACH technique-based particle swarm optimization. *IEEE Access* **2022**, *10*, 36935–36948. [\[CrossRef\]](#)
42. Haris, M.; Bhatti, D.M.S.; Nam, H. A Fast-Convergent Hyperbolic Tangent PSO Algorithm for UAVs Path Planning. *IEEE Open J. Veh. Technol.* **2024**, *5*, 681–694. [\[CrossRef\]](#)

43. Li, D.; bin Wen, X. An Improved PSO Algorithm for Distributed Localization in Wireless Sensor Networks. *Int. J. Distrib. Sens. Netw.* **2015**, *11*, 970272. [\[CrossRef\]](#)
44. Meng, Y.; Zhi, Q.; Zhang, Q.; Yao, N. A Two-Stage Particle Swarm Optimization Algorithm for Wireless Sensor Nodes Localization in Concave Regions. *Information* **2020**, *11*, 488. [\[CrossRef\]](#)
45. Kim, Y.G.; Lee, M.J. Scheduling multi-channel and multi-timeslot in time-constrained wireless sensor networks via simulated annealing and particle swarm optimization. *IEEE Commun. Mag.* **2014**, *52*, 122–129. [\[CrossRef\]](#)
46. Gupta, N.; Kumar, N.; Jain, S. Coverage problem in wireless sensor networks: A survey. In Proceedings of the 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPEs), Paralakhemundi, India, 3–5 October 2016; pp. 1742–1749.
47. Amutha, J.; Sharma, S.; Sharma, S.K. Strategies based on various aspects of clustering in wireless sensor networks using classical, optimization and machine learning techniques: Review, taxonomy, research findings, challenges and future directions. *Comput. Sci. Rev.* **2021**, *40*, 100376. [\[CrossRef\]](#)
48. Sheng, Z.; Mahapatra, C.; Leung, V.C.; Chen, M.; Sahu, P.K. Energy efficient cooperative computing in mobile wireless sensor networks. *IEEE Trans. Cloud Comput.* **2015**, *6*, 114–126. [\[CrossRef\]](#)
49. Hacıoglu, G.; Kand, V.F.A.; Sesli, E. Multi objective clustering for wireless sensor networks. *Expert Syst. Appl.* **2016**, *59*, 86–100. [\[CrossRef\]](#)
50. Elshakhs, Y.S.; Deliparaschos, K.M.; Charalambous, T.; Oliva, G.; Zolotas, A. A comprehensive survey on Delaunay triangulation: Applications, algorithms, and implementations over CPUs, GPUs, and FPGAs. *IEEE Access* **2024**, *12*, 12562–12585. [\[CrossRef\]](#)
51. Thomas, D.; Shankaran, R.; Sheng, Q.Z.; Orgun, M.A.; Hitchens, M.; Masud, M.; Ni, W.; Mukhopadhyay, S.C.; Piran, M.J. QoS-aware energy management and node scheduling schemes for sensor network-based surveillance applications. *IEEE Access* **2020**, *9*, 3065–3096. [\[CrossRef\]](#)
52. Samadi, R.; Nazari, A.; Seitz, J. Intelligent energy-aware routing protocol in mobile IoT networks based on SDN. *IEEE Trans. Green Commun. Netw.* **2023**, *7*, 2093–2103. [\[CrossRef\]](#)
53. Abbas, O.K.; Abdullah, F.; Radzi, N.A.M.; Salman, A.D.; Abdulkadir, S.J. Survey on Clustered Routing Protocols Adaptivity for Fire Incidents: Architecture Challenges, Data Losing, and Recommended Solutions. *IEEE Access* **2024**, *12*, 113518–113552. [\[CrossRef\]](#)
54. Masood Ahmed, K.; Shams, R.; Khan, F.H.; Luque-Nieto, M.Á. Securing Underwater Wireless Sensor Networks: A Review of Attacks and Mitigation Techniques. *IEEE Access* **2024**, *12*, 161096–161133. [\[CrossRef\]](#)
55. Shi, Y.; Eberhart, R. A modified particle swarm optimizer. In Proceedings of the 1998 IEEE International Conference on Evolutionary Computation Proceedings, IEEE World Congress on Computational Intelligence (Cat. No.98TH8360), Anchorage, AK, USA, 4–9 May 1998; pp. 69–73. [\[CrossRef\]](#)
56. Haris, M.; Bhatti, D.M.S.; Nam, H. An Improved Particle Swarm Optimization Algorithm Based on S-shaped Activation Function for Fast Convergence. In Proceedings of the 2022 13th International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Republic of Korea, 19–21 October 2022; pp. 239–243.
57. Chen, H.L.; Yang, B.; Wang, S.J.; Wang, G.; Liu, D.Y.; Li, H.Z.; Liu, W.B. Towards an optimal support vector machine classifier using a parallel particle swarm optimization strategy. *Appl. Math. Comput.* **2014**, *239*, 180–197. [\[CrossRef\]](#)
58. Yao, X.; Liu, Y.; Lin, G. Evolutionary programming made faster. *IEEE Trans. Evol. Comput.* **1999**, *3*, 82–102. [\[CrossRef\]](#)
59. Ghorpade, S.N.; Zennaro, M.; Chaudhari, B.S.; Saeed, R.A.; Alhumyani, H.; Abdel-Khalek, S. Enhanced Differential Crossover and Quantum Particle Swarm Optimization for IoT Applications. *IEEE Access* **2021**, *9*, 93831–93846. [\[CrossRef\]](#)
60. Hu, M.; Wu, T.; Weir, J.D. An Adaptive Particle Swarm Optimization With Multiple Adaptive Methods. *IEEE Trans. Evol. Comput.* **2013**, *17*, 705–720. [\[CrossRef\]](#)
61. Lin, Z. Optimizing Kernel Extreme Learning Machine based on a Enhanced Adaptive Whale Optimization Algorithm for classification task. *PLoS ONE* **2025**, *20*, e0309741. [\[CrossRef\]](#)
62. Chen, M.; Wang, H.; Chen, X. Path planning optimization of medical service robots based on PSO. In Proceedings of the 2022 6th International Conference on Wireless Communications and Applications (ICWCAPP), Haikou, China, 20–21 August 2022; pp. 139–141. [\[CrossRef\]](#)
63. Han, W.; Yang, P.; Ren, H.; Sun, J. Comparison study of several kinds of inertia weights for PSO. In Proceedings of the 2010 IEEE International Conference on Progress in Informatics and Computing, Shanghai, China, 10–12 December 2010; Volume 1, pp. 280–284. [\[CrossRef\]](#)
64. Liu, W.; Wang, Z.; Yuan, Y.; Zeng, N.; Hone, K.; Liu, X. A Novel Sigmoid-Function-Based Adaptive Weighted Particle Swarm Optimizer. *IEEE Trans. Cybern.* **2021**, *51*, 1085–1093. [\[CrossRef\]](#) [\[PubMed\]](#)
65. Xu, L.; Song, B.; Cao, M. An improved particle swarm optimization algorithm with adaptive weighted delay velocity. *Syst. Sci. Control Eng.* **2021**, *9*, 188–197. [\[CrossRef\]](#)
66. Sharmin, S.; Ahmedy, I.; Md Noor, R. An energy-efficient data aggregation clustering algorithm for wireless sensor Networks using hybrid PSO. *Energies* **2023**, *16*, 2487. [\[CrossRef\]](#)

67. Tian, P.; Li, K.; Wu, X.; Yu, S.; Hu, Y.; Shi, Q. Indoor Geomagnetic Matching Location Based on Iterative Local Search and Improved Particle Swarm Fusion. *IEEE Robot. Autom. Lett.* **2025**, *10*, 5337–5344. [[CrossRef](#)]
68. Rahnemay, M.; Farzinvash, L.; Zolfi, M.; Taherkordi, A. ECMSH: An Energy-efficient and Cost-effective data harvesting protocol for Mobile Sink-based Heterogeneous WSNs using PSO-TVAC. *Ad Hoc Netw.* **2024**, *164*, 103629. [[CrossRef](#)]
69. Hu, Y.; Zhang, Y.; Gong, D.; Sun, X. Multiparticipant Federated Feature Selection Algorithm With Particle Swarm Optimization for Imbalanced Data Under Privacy Protection. *IEEE Trans. Artif. Intell.* **2023**, *4*, 1002–1016. [[CrossRef](#)]
70. Zhang, L. A deployment and coverage optimization algorithm for self-powered wireless sensor networks based on hybrid swarm intelligence. *IEEE Sens. J.* **2022**, *23*, 20705–20714. [[CrossRef](#)]
71. Ou, Y.; Qin, F.; Zhou, K.Q.; Yin, P.F.; Mo, L.P.; Mohd Zain, A. An improved grey wolf optimizer with multi-strategies coverage in wireless sensor networks. *Symmetry* **2024**, *16*, 286. [[CrossRef](#)]
72. Song, J.; Hu, Y.; Luo, Y. Wireless Sensor Network Coverage Optimization Based on the Novel Enhanced Hunter–Prey Optimization Algorithm. *IEEE Sens. J.* **2024**, *24*, 31172–31187. [[CrossRef](#)]
73. Yang, S.Y.; Xiang, Y.H.; Kang, D.W.; Zhou, K.Q. An Improved Cuckoo Search Algorithm for Maximizing the Coverage Range of Wireless Sensor Networks. *Baghdad Sci. J.* **2024**, *21*, 0568. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.