A Novel Exponential Dynamic Inertia Weight for Particle Swarm Optimization

Abstract—**The traditional particle swarm optimization (PSO) algorithm suffers from shortcomings like easily falling into local optimum and inadequate sharing of information among particles, To ad-dress these limitations and enhance the search capacity of the particle swarm algorithm, we present a novel particle swarm optimization algorithm known as Exponential Dynamic Inertia Weight for Particle Swarm Optimization (ExDyPSO) in this paper. ExDyPSO is composed of two parts: firstly, by introducing dynamic inertia weight based on exponential distributions and acceleration factors that vary with the number of iterations, harmonizes the global and local search capabilities. Secondly, a stochastic particle-based jump-out strategy is proposed to surmount the case of particles falling into stagnation during the search process, thus effectively addressing the issue that PSO is prone to falling into local optimum. To assess the performance of ExDyPSO, we carried out experiments on eight benchmark functions and compared its performance against four alternative PSO variations. The experimental findings demonstrate that ExDyPSO achieves quicker and more accurate convergence towards the global optimal solution, all the while sustaining population diversity.**

Keywords—particle swarm optimization (PSO), dynamic inertia weights, jump-out strategy, global search, local search.

I. INTRODUCTION

The Particle Swarm Optimization (PSO) algorithm is a stochastic optimization method whose fundamental principles are inspired by collective behaviors observed in groups of organisms, like bird flocking and fish schooling dynamics [1]. In PSO, each particle in the group represents a potential solution to the optimization problem. Initially, particles are dispersed across the search domain with velocities chosen randomly. Subsequently, each particle updates its speed based on both its individual best experience and the collective best experience of the entire population. The unique interaction mechanism between particles in the PSO algorithm enable them to efficiently navigate toward the optimal position under reasonable guidance. As a result, the PSO algorithm finds applications in various domains, such as parametric valuation [2], neural networks [3], image processing [4], and signal processing [5].

Inertia weight and acceleration factors are essential parameters within the PSO algorithm, playing an important role in the velocity update process. Inertia weight serves to retain each particle's prior motion direction. If the inertia weight's value is substantial, particles tend to persist in their former direction, enhancing algorithm convergence speed. Yet, when this weight is minimal, particles assign minimal significance to their previous motion direction. Furthermore, the acceleration factors also significantly impact the PSO algorithm's performance.

In recent years, inertia weight and acceleration factors have been extensively investigated to achieve enhanced optimization accuracy and accelerated convergence [6],[7],[21]. Drawing on prior research, we present a novel particle swarm optimization algorithm known as Exponential Dynamic inertia weight for Particle Swarm Optimization (ExDyPSO). The algorithm introduces dynamic inertia weight based on exponential distribution, along with an acceleration factor that adapts to the iteration count. This design aims to attain a balance between exploitation and exploration. Furthermore, to overcome the stagnation phenomenon, we devise a jump-out strategy to help particles get rid of the local optimal attraction by selecting random particles to participate in the search.

To assess the effectiveness and optimization performance of the ExDyPSO, we conducted tests across eight benchmark functions and conducted comparisons with four other advanced PSO variants. The experimental outcomes reveal that ExDyPSO attains a commendable balance between global and local search, while manifesting steady performance concerning solution accuracy and convergence velocity. Furthermore, the analysis of population diversity shows that ExDyPSO also demonstrates effectiveness in adjusting variety.

The remainder of this paper is structured as follows. Section 2 discusses prevalent PSO variations while Section 3 details the proposed ExDyPSO algorithm. Section 4 presents the experimental results and corresponding analyses to demonstrate the effectiveness of ExDyPSO. Finally, we conclude and share future perspectives in Section 5.

II. RELATED WORKS

PSO, an optimization algorithm pioneered by Kennedy and Eberhart [1], emulates the conduct of a population, similar to a bird flock or fish school, with each particle symbolizing a potential solution to some optimization problem. A particle denoted by i $(1 \le i \le N)$ possesses two vectors: a position vector $X_i = [X_{i,1}, X_{i,2}, \ldots, X_{i,D}]$ and a velocity vector $V_i = [V_{i,1}, V_{i,2}, \ldots, V_{i,D}]$. These vectors are

initialized randomly. The regulations governing the update of particle velocity and position are illustrated below:

$$V_{i,d}^{t+1} = wV_{i,d}^{t} + c_1 r_1 \left(Pbest_{i,d}^{t} - V_{i,d}^{t}\right) + c_2 r_2 \left(gbest_{d} - V_{i,d}^{t}\right) \quad (1)$$

$$X_{i,d}^{t+1} = X_{i,d}^{t} + V_{i,d}^{t+1} \quad 1 \le i \le N, 1 \le d \le D \quad (2)$$

In Eqs. (1) and (2), symbols $X_{i,d}^{t}$, and $V_{i,d}^{t}$ represent the position and velocity of particle i in the d^{th} dimension at iteration t, respectively. Here, w symbolizes the inertia weights, while c_1 and c_2 are the acceleration factors, all set to 2. 0 [1]. Similarly, r_1 and r_2 denote two random numbers within the interval [0, 1]. Pbest_i designates the individual best position found by particle i, while gbest signifies the global best position found by all particles so far. The velocity and position of the particles will be updated according to Eq. (1) and updated iteratively through Eq. (2) to gradually search for and approach the optimal solution.

Numerous improvement methods have been suggested to further improve the performance of traditional PSO. Among them, Shi and Eberhart [8] introduced an enhanced approach named PSO-LDIW to optimize the PSO algorithm by linearly reducing the variation of inertia weight w with the number of iterations. The up-date rule for the inertia weight (w) is as follows:

$$w = (w_1 - w_2) \times \frac{t_{max} - t}{t_{max}} + w_2 \quad (3)$$

Here, w_1 represents the initial value of the inertia weight, while w_2 signifies its final value. In the beginning, a high inertia weight helps to find particles with favourable performance. As the process advances, a lower inertia weight becomes more conducive to precise exploration. The parameter t represents the current number of iterations, while t_{max} indicates the maximum allowable iterations. Additionally, numerous algorithms have been proposed for improving the inertia weight w. Examples include linear decreasing w [9], stochastic w [10], and chaotic dynamics w [11].

It has been recognized that the performance of the algorithm is influenced by the variability of the acceleration factor. Consequently, researchers have pursued numerous enhancements to the acceleration factor. For instance, a PSO algorithm incorporating a time-varying acceleration coefficient (PSO-TVAC) was introduced [6]. Evolutionary factors have been defined and population evolutionary states categorized into four scenarios: exploration, exploitation, escape, and convergence [11]. Furthermore, a PSO algorithm with a positive cosine acceleration factor has also been proposed [12].

Apart from parameter enhancements, certain researchers have directed their attention toward formulating diverse topologies to amplify the efficiency of the PSO algorithm. The SPSO algorithm enhances convergence speed by dynamically updating acceleration coefficients based on distinct evolutionary states [13]. Conversely, the Competitive Particle Swarm Optimizer (CSO) algorithm was devised to address extensive optimization challenges [14]. This involves the implementation of a pairwise competition mechanism. Through this mechanism, the particles that lose the match are adjusted to the position of the winning particles. Moreover, introducing time delays into PSO algorithms to mitigate local optima has been proposed [15]. The Multimodal Delayed Particle Swarm Optimization (MDPSO) algorithm, by evaluating evolutionary factors during each iteration, partitions the particle swarm's evolutionary state equidistantly [16]. A sorted particle swarm with mixed paradigms has been proposed to improve the optimization performance; moreover, novel adaptation schemes both for the ratio of each paradigm and the constriction coefficients are proposed during the iteration[20]. Several academic researchers have developed a flexible software framework for PSO, called PSO-X, which is specifically designed to integrate the use of automatic configuration tools into the process of generating PSO algorithms [22].

While the aforementioned improvement methods enhance various aspects of traditional PSO algorithms, they also exhibit certain drawbacks and limitations. Firstly, the strategy of amalgamating diverse evolutionary algorithms augments algorithmic intricacy. The amalgamation of distinct algorithms necessitates judicious parameter configuration and intricate operational procedures, thereby escalating the challenge of algorithm implementation and debugging. Secondly, certain improvement methods may negatively affect the convergence performance of the algorithm. Although these methods may bolster global search potential and diversity, they might occasionally result in slower convergence or local optima. Building upon the aforementioned shortcomings and restrictions, we devise a novel algorithm aims to address the shortcomings of the traditional PSO algorithm in terms of reduced population diversity and the local optimal solution problem.

III. Exponential Dynamic Inertia Weight

In this section, we present our algorithm, focusing on three aspects: (A) for enhancing the inertia weight, we propose a dynamic inertia weight that follows an exponential distribution; (B) In order to achieve a balance between global and local search capabilities, we propose a linear adjustment strategy for the acceleration factor; and, (C) we propose a jump-out strategy based on random particles to avoid falling into local optimal solutions.

A. Exponential Dynamic Inertia Weight Strategy

Within the conventional particle swarm optimization algorithm, the inertia weight (w) embodies the influence of historical velocity on the present pace and ascertains the equilibrium between the algorithm's global and local search competencies. With a higher inertia weight (w), particles prioritize global search, whereas a lower (w) emphasizes local exploration. Therefore, for achieving an improved equilibrium between global and local search strategies, we propose a method named dynamic inertia weight based on exponential distribution.

A non-linear diminishing strategy has been introduced to enhance algorithmic performance by adapting the inertia weight, particularly to mitigate issues of premature convergence. Despite the improved methods achieved by these enhanced approaches, they may give rise to challenges such as inadequate population diversity and decelerated convergence during the later stages of the algorithm's execution. To harmonize the algorithm's global and local search capacities, we propose a dynamic adjustment strategy based on the exponential distribution.

The exponential distribution is a continuous probability distribution defined over the range [0, +∞], which has critical applications in fields such as multivariate statistical analysis

and mathematical statistics. Let the random variable x has a density function as depicted in Equation (4):

$$f(x) = \begin{cases} \frac{1}{\theta} e^{-\frac{x}{\theta}}, x > 0 \ (\theta > 0) \\ 0, x \leq 0 \end{cases} \quad (4)$$

Then x is said to follow an exponential distribution with parameter θ , denoted as x ~ Exp (θ) . The distribution function is presented as follows:

$$F(x) = \begin{cases} 1 - e^{-\theta x}, x \geq 0 \\ 0, x < 0 \end{cases} (\theta > 0) \quad (5)$$

The expression for the inertia weight, following the incorporation of the exponential distribution, is as follows:

$$w = w_{max} - (w_{max} - w_{min})\left(\frac{t}{t_{max}}\right)^2 + (-1)^{\text{rand}()}\lambda(X \sim Exp(\theta)) \quad (6)$$

Where w_{max} signifies the maximum value of the manually entered inertia weight w, and w_{min} represents its lower limit, t denotes the current iteration count, t_{max} signifies the maximum iteration count, and $rand()$ corresponds to a random parity value. The exponential distribution with parameter θ is denoted by $X \sim Exp(\theta)$ and λ is a vital parameter that impacts the range of fluctuation of the inertia weight (w). This form of formulation is designed to take full advantage of the properties of the exponential distribution in order to balance the capabilities of global exploration and local search, thus improving the performance of the algorithm. By employing this strategy, we are better able to cope with the complexity of the problem.

To ensure w varies within the intended range, selecting an appropriate value is necessary. In this context, a range of values for λ was tested, viz. , λ={0. 05, 0. 1, 0. 15, 0. 2, 0. 25, 0. 3, 0. 35, 0. 4, 0. 45, 0. 5} (comprising 10 instances), and it was observed that a value of 0. 2 achieved the intended outcome. As a result, for the subsequent experiments, we set the parameter λ to 0. 2. Fig. 1 shows the line graph depicting the dynamic inertia weight (w) in relation to iteration count, employing the parameter values w_{max}= 1. 0, w_{min}= 0. 4, and t_{max} = 400.

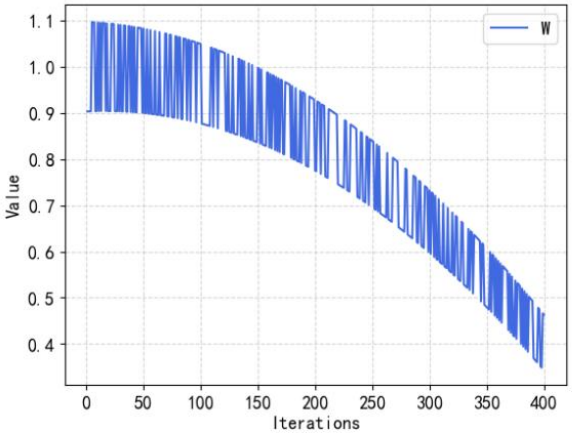

Fig. 1. Dynamic inertia weight based on exponential distribution.

From Eq. (6) and the insights from Fig. 1, we can conclude that using the dy-namic adjustment strategy based on exponential distribution makes the inertia weight show a nonlinear change throughout the search process and gradually decreases. During the initial phase of the search, the inertia weight changes by a large margin, which facilitates the algorithm to search in the broader solution space, thus improving the diversity of the particle population. Simultaneously, during the later search phases, the algorithm can also obtain diverse inertia weight values, thereby further heightening the search precision of the algorithm. The incorporation of a dynamic adjustment strategy based on the exponential distribution enables the harmonization of the algorithm's global and local search potentials, thereby mitigating the limitations in the traditional PSO algorithm during both its initial and concluding search phases.

B. Dynamic Acceleration Factors Based on the Number of Iterations

In traditional PSO algorithms, acceleration factors are typically set to fixed values, often as *c1=c2=2.0* [1]. However, studies have demonstrated that the selection of acceleration factors impacts the algorithm's performance. A higher c1 acceleration factor directs particles to focus more on their historical optimal solutions, thus enhancing local search capabilities. However, this could lead to the algorithm becoming overly dispersed during the search process, rendering the convergence towards the global optimal solution challenging. Conversely, a higher c2 acceleration factor compels particles to concentrate more on the global optimal solution, thereby accelerating the global search capability. However, this may result in the algorithm excessively converging towards the local optimal solution. In order to achieve a balance between global and local search capabilities, we propose a linear adjustment strategy for the acceleration factor, which is dependent on the algorithm's iteration count. This strategy enables the acceleration factor to dynamically adjust throughout the search process. The exact calculation formula is presented below:

$$c_1 = c_{max} - (c_{max} - c_{min}) \times \left(\frac{t}{t_{max}}\right)^2 \quad (7)$$

$$c_2 = c_{min} + (c_{max} - c_{min}) \times \left(\frac{t}{t_{max}}\right)^2 \quad (8)$$

where c_{max} and c_{min} represent the initial maximum and minimum acceleration factors, while t_{max} and t denote the maximum and current iterations of the algorithm, respectively. Within the ExDyPSO algorithm, we incorporate varying particle learning strategies at distinct stages through the generation of dynamic acceleration factors. This design enhances particle focus on self-learning during the search's initial stages, thereby fostering improved optimization throughout the search space. Furthermore, during the algorithm's later stages, we fine-tune the acceleration factor to prioritize population learning. This implies that the particles allocate greater attention to the global optimal solution, thus expediting the algorithm's convergence. Through this strategy, we achieve a balance between particle self-learning and population learning, consequently enhancing the algorithm's search performance.

C. Jump-Out Strategy

Sometimes, a particle might become trapped in a local optimum, halting the particle's fitness updates after a limited number of iterations. Falling into the local optimum deprives the particle of the chance to explore the unexplored solution space. This algorithm introduces a stochastic particle-based

jump-out strategy designed to aid particles in escaping from the local optimal situation. When a particle stops updating after G successive iterations (where the fitness ceases to change), the jump-out strategy will be invoked to enhance the particle's exploration capability.

$$X_{\text{new}} = (1 - r_3) \cdot Pbest_{i,d} + r_3 \cdot Pbest_{j,d} + r_4 \cdot \left(Pbest_{i,d} - Pbest_{j,d}\right) \quad 1 \le i, j \le N \tag{9}$$

Where X_{new} represents the new candidate position of particle *i*, with *i* and *j* being randomly chosen particles, while r_3 is a random number within the interval [0,1] and r_4 is a random number between [-1,1]. Through the incorporation of these stochastic factors, the particles are infused with a degree of randomness and diversity. The position of X_i is substituted only under the condition that the fitness value of X_{new} surpasses that of the present particle X_i.

The proposed jump-out strategy exhibits the subsequent characteristics. Firstly, it provides candidate positions for particles in a stagnant state, thus offering them the possibility of jumping out of the deep local optimum. Doing so can help these particles escape the predicament of being trapped in a local optimum. In this regard, we are more focused on helping particles that are experiencing difficulties than the previous jump-out strategy. Additionally, the strategy fully capitalizes on the information exchange among particles by improving the positions of random particles to be provided to the current particle. This exchange of information serves to augment the search capability of the entire particle swarm, stimulating the particles to explore the global search space more effectively. With these key features, our jump-out strategy shows significant advantages over traditional methods in solving optimization problems.

D. ExDyPSO Algorithm

Figure 2 illustrates the flowchart and steps of ExDyPSO.

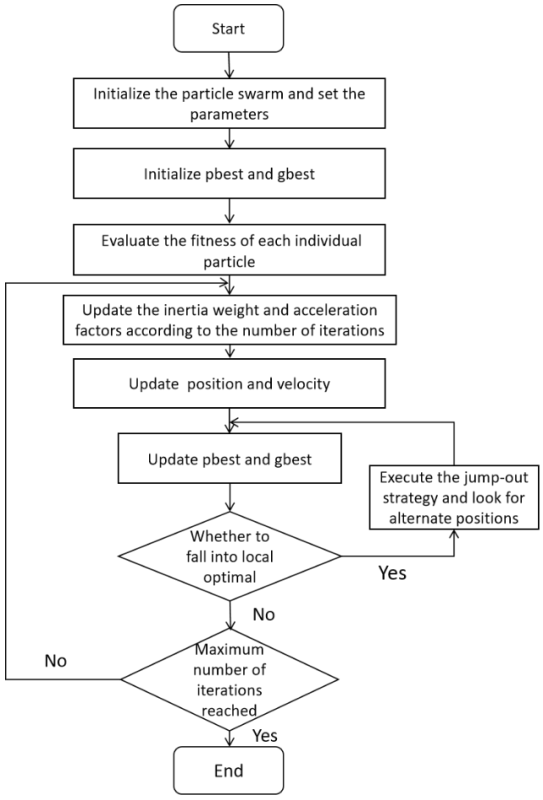

Fig. 2. Flowchart of ExDyPSO algorithm.

Step 1. Randomly initialize the positions and velocities of all particles.

Step 2. Calculates the adaptation value of each particle and initialize Pbest.

Step 3. Updates the inertia weight and acceleration factors using Eqs. (2), (3), and (4).

Step 4. Updates the particles' velocity and position.

Step 5. Computes the fitness values for all particles in the population and update Pbest.

Step 6. Determines if the particle is trapped in a local optimum. If so, apply the jump-out strategy using Eq. (5).

Step 7. Checks whether the ExDyPSO algorithm meets the termination condition. If satisfied, ExDyPSO produces the optimal solution; otherwise, proceed back to step 3.

IV. Using the Template Experimental Validation

This section describes validation of the proposed ExDyPSO algorithm. We initialize each PSO algorithm using 60 particles and execute independently on all benchmark functions 50 times. The experimental environment is configured as follows: host hardware configuration Intel(R) Core(TM) i7-1065G7 CPU @ 1. 30GHz 1. 50 GHz, with 16. 0 GB of RAM (15. 8 GB available), and the operating system is Windows 10 64-bit Home Edition. All code in the experiment was writ-ten in Python, version number 3. 10. 1, and the experimental platform was the lightweight PyCharm Community.

A. Benchmark Functions

We employ eight test functions to investigate the performance of the ExDyPSO algorithm, as shown in Table 1. F_1 and F_2 are single-peak, and the rest are multi-peak optimization problems. Among them, the single-peak function F_2, along with the multi-peak functions F_3, F_4, F_5, F_6, and F_8 have an asymmetric environment. Each of the chosen functions represents a minimization problem. Table I presents the minimum and search range for each function, and the problem's dimensionality D. Four statistical metrics are used to assess performance, viz. , mean ranking, standard deviation, minimum and average.

TABLE I. Configuration of Benchmark Functions.

Function Number	Function Name	Function Expression	Search Range	Minimum
$F_1(x)$	Sphere	$\sum_{i=1}^{D} x_i^2$	$[-100,100]^D$	0
$F_2(x)$	Sehwwefel P2.21	$\max_{1<i<D} \lvert x_i \rvert$	$[-100,100]^D$	0
$F_3(x)$	Rastrigin	$\sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	$[-5.12,5.12]^D$	0
$F_4(x)$	Schwefel P1.2	$\sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j\right)^2$	$[-100,100]^D$	0
$F_5(x)$	Griewank	$\frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \left[\cos\left(\frac{x_i}{\sqrt{i}}\right)\right] + 1$	$[-600,600]^D$	0
$F_6(x)$	Rosenbrock	$\sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$	$[-2,2]^D$	0
$F_7(x)$	Sum Squares	$\sum_{i=1}^{D} i x_i^2$	$[-10,10]^D$	0
$F_8(x)$	Alpine	$\sum_{i=1}^{D} \lvert x_i \sin(x_i) + 0.1x_i \rvert$	$[-10,10]^D$	0

B. Parameter Sensitivity Analysis

Within the ExDyPSO algorithm, three critical parameters exist: w_{max} , w_{min} , and the count of jump-out strategy evaluations, denoted as G. To comprehend the influence of these parameters on ExDyPSO's performance, a parameter sensitivity analysis will be conducted in this subsection.

The variabl w_{max} es and w_{min} respectively denote the upper and lower bounds of the inertia weight. To enhance early-stage exploration performance and expedite later-stage convergence, the inertia weight value should progressively decrease as iterations increase. The candidate positions for $[w_{max}, w_{min}]$ are chosen as {[0. 9,0. 3], [0. 9,0. 4], [0. 9,0. 5]……[1. 1, 0. 5], [1. 1, 0. 6]} (comprising 9 instances) for testing. If the number of iterations G is set to a smaller value, particles exhibit higher search velocity during the exploration phase; however, their capacity to locate the optimal fitness is compromised. Conversely, when G is set to a larger value, particles are susceptible to falling into local optima. To ensure the algorithm executes the jump-out strategy within a reasonable iteration count, this study defines the range of G values as {5, 6, 7, . . . , 14, 15} (com prising 11 instances).

The optimization scores are visualized through 3D bar charts, as shown in Fig. 3, wherein the bar height signifies the extent of superior performance. Fig. 3 reveals the notable influence of various [w_{max} , w_{min}] and G values on ExDyPSO, and notably, the histogram reaches its maximum height, signifying optimal performance, when [w_{max} , w_{min}]=[1. 0,0. 5] and G=11. Consequently, the hyperparameters in ExDyPSO are configured as w_{max}=1. 0, w_{min}= 0. 5, and G=11.

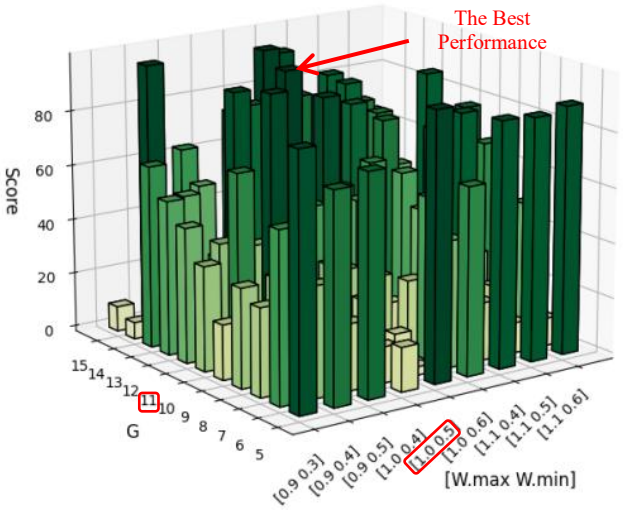

Fig. 3. Performance presentation of differ-ent parameter combinations.

C. Diversity Analysis Performance Comparison

The variation in population diversity throughout the iterative process serves as an intuitive indicator of optimization performance. In this segment, we undertake comparative experiments to assess population diversity. The diversity of the population at the k-th iteration is computed using the following formula [17],

$$S(k) = \frac{1}{M}\sum_{i=1}^{M}\sqrt{\sum_{j=1}^{D}\left(x_{ij}(k) - \bar{x}_j(k)\right)^2} \qquad (6)$$

Here, M represents the population size, D specifies the dimension of the optimization problem, x_{ij} corresponds to the i-th particle in the j-th dimension, and $\bar{x}_j(k)$ denotes the mean of all particles in the j-th dimension during the k-th iteration, expressed as $\bar{x}_j(k) = (1/M)\sum_{i=1}^{M} x_{ij}(k)$. In this segment, we utilize the experimental outcomes of the Rastrigin function within a 30-dimensional space as an illustration to depict the changes in population diversity.

As shown in Fig. 4, a noticeable trend emerges: the diversity of populations in both the conventional particle swarm algorithm and the ExDyPSO algorithm diminishes progressively with an increase in iterations. Reduced population diversity indicates a gradual concentration of the population toward a particular region within the search space. Notably, the ExDyPSO algorithm exhibits superior convergence in the later phases of the optimization process compared to the classical PSO algorithm. Consequently, this indicates that ExDyPSO exhibits improved capability in converging towards the global optimal solution, manifesting more effective convergence performance throughout the optimization process compared to the conventional PSO algorithm.

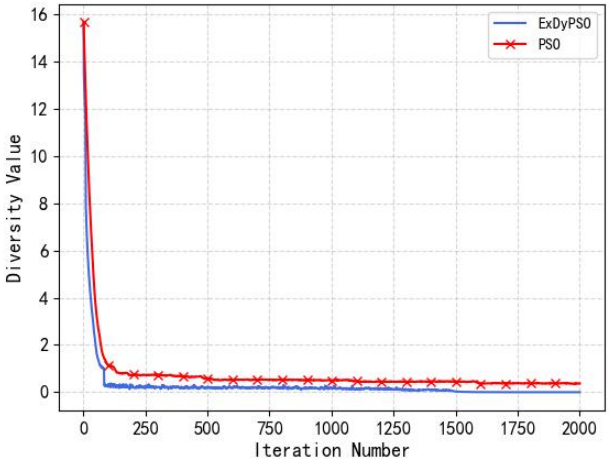

Fig. 4. Diversity measure for Rastrigin function F3(x).

D. Performance Comparison

We compare the proposed ExDyPSO algorithm with four state-of-the-art PSO variants, viz. , a multiple adaptive strategies-based particle swarm optimization algorithm (MAPSO) algorithm [18], a novel multimodal delayed particle swarm optimization (MDPSO) algorithm [16], a particle swarm optimization algorithm combined with time-varying acceleration coefficients (PSO-TVAC) [6] and the Quantum Delta-Potential-Well-based Particle Swarm Optimization (QDPSO) algorithm [19]. Tables Ⅱ and III present a comparison of ExDyPSO results against the other four PSO algorithms across eight test functions in both 30 and 60 dimensions. The optimal outcomes among these eight test functions are shown in bold.

From the outcomes presented in Table Ⅱ , ExDyPSO achieves the highest performance on seven out of the eight test functions. The single-peaked functions (F1 and F2) have a simple shape and a single optimal solution. These functions are typically employed to assess the global search capacity and convergence performance of an optimization algorithm, i. e. , the exploitation ability of the algorithm. Given that ExDyPSO incorporates a dynamic adaptation mechanism it can deal with such functions well, and consistently excels in its performance for the single-peak functions F1 and F2.

Multi-peak test functions are characterized by having numerous local optima within the search space. These functions exhibit intricate forms and multiple local optima and are commonly employed to assess optimization

algorithms' proficiency in addressing multimodal problems. Among the multi-peak functions (F3~F8), ExDyPSO is excellent in performance across the functions except for F5. Despite that, it finds the smallest minima. The complexity of the F5 function lies in its highly nonlinear shape and the relatively short distance between local optimal solutions. This structure increases the difficulty of finding a globally optimal solution on this function and poses a higher challenge to the optimization algorithm. QDPSO exhibits superior relative performance due to the so-called quantum Delta potential well model.

TABLE II. EXPERIMENTAL RESULTS OF PSO ALGORITHMS FOR 30-DIMENSIONAL TEST FUNCTIONS.

Function Number	Criteria	ExDyPSO	MDPSO	PSO-TVAC	MAPSO	QDPSO
$F_1(x)$	Minimum	**1.595E-21**	5.537E+00	5.335E-06	8.726E-09	6.402E-20
	Mean	**3.364E+02**	1.176E+03	9.052E+02	1.354E+03	6.763E+02
	Std	2.445E+03	5.106E+03	5.308E+03	**1.483E+03**	4.064E+03
	Rank	**1**	4	2	5	3
$F_2(x)$	Minimum	**5.618E-03**	7.377E+00	6.861E-02	1.451E+00	2.093E-01
	Mean	**1.495E+00**	1.151E+01	2.985E+00	1.526E+01	1.385E+01
	Std	**6.422E+00**	9.794E+00	1.016E+01	6.992E+00	1.977E+01
	Rank	**1**	3	2	5	4
$F_3(x)$	Minimum	**3.024E+01**	6.971E+01	3.916E+01	4.816E+01	3.070E+01
	Mean	**3.562E+01**	1.108E+02	6.319E+01	1.647E+02	5.173E+01
	Std	**2.798E+01**	5.840E+01	4.764E+01	4.697E+01	5.241E+01
	Rank	**1**	4	3	5	2
$F_4(x)$	Minimum	**1.812E-02**	3.178E+02	1.108E-01	1.233E+02	6.982E+01
	Mean	**9.913E+02**	7.753E+03	2.213E+03	4.306E+03	5.507E+03
	Std	5.746E+03	1.328E+04	1.083E+04	**2.888E+03**	1.118E+04
	Rank	**1**	5	2	3	4
$F_5(x)$	Minimum	**1.102E-02**	8.736E-01	1.381E-02	8.514E-01	1.631E-02
	Mean	1.626E+01	1.148E+01	4.895E+01	1.288E+01	**6.347E+00**
	Std	3.808E+01	4.720E+01	1.097E+02	**1.333E+01**	3.745E+01
	Rank	4	**2**	5	3	**1**
$F_6(x)$	Minimum	**2.128E+01**	2.809E+01	2.532E+01	2.346E+01	2.845E+01
	Mean	**3.414E+01**	1.066E+02	4.560E+01	8.555E+01	1.099E+02
	Std	1.206E+02	4.874E+02	1.657E+02	**6.470E+01**	3.939E+02
	Rank	**1**	4	2	3	5
$F_7(x)$	Minimum	**3.082E-20**	1.493E+00	8.439E-07	1.152E-09	5.988E-19
	Mean	**4.517E+01**	2.024E+02	1.354E+02	1.748E+02	8.925E+01
	Std	3.434E+02	7.590E+02	8.003E+02	**2.032E+02**	5.656E+02
	Rank	**1**	5	3	4	2
$F_8(x)$	Minimum	**3.413E-11**	6.420E+00	7.595E-05	3.907E-01	4.167E-08
	Mean	**1.274E+00**	1.064E+01	2.344E+00	9.380E+00	2.420E+00
	Std	**4.921E+00**	7.378E+00	7.938E+00	6.316E+00	7.749E+00
	Rank	**1**	5	2	4	3

To assess ExDyPSO's performance on prob-lems with high dimensionality, eight test functions are applied to a problem dimension D of 60, and results shown in Table III.

When the problem dimension D is high, at 60, it becomes evident that ExDyPSO performs well across all eight test functions, securing the top position in six of them. These encompass the single-peak function F1, along with the multi-peak functions F3, F4, F6, F7, and F8. This underscores ExDyPSO's strong optimization capability when addressing high-dimensional problems. The F2 function has a highly nonlinear shape and a complex local optimal solution structure in high dimensions which the ExDyPSO failed to achieve the highest score, but its comprehensive assessment metric still surpasses that of MAPSO and MDPSO, placing it in second place. Within the F2 test function, PSO-TVAC performs better, attributed to the incorporation of its algorithm's time-varying acceleration coefficient. ExDyPSO achieves commendable results in the F5 test function, like in F2, securing the second position in the comprehensive performance assessment despite not attaining the highest mean value.

TABLE III. EXPERIMENTAL RESULTS OF PSO ALGORITHMS FOR 60-DIMENSIONAL TEST FUNCTIONS.

Function Number	Criteria	ExDyPSO	MDPSO	PSO-TVAC	MAPSO	QDPSO
$F_1(x)$	Minimum	**1.353E-08**	2.309E+02	3.051E-02	5.874E+02	1.373E-06
	Mean	**8.400E+02**	4.213E+03	2.227E+03	5.772E+03	1.409E+04
	Std	5.473E+03	1.256E+04	1.269E+04	**2.315E+03**	3.279E+04
	Rank	**1**	3	2	4	5
$F_2(x)$	Minimum	1.309E+00	1.273E+01	**4.699E-01**	2.488E+01	1.610E+01
	Mean	5.549E+00	1.797E+01	**5.031E+00**	2.899E+01	4.673E+01
	Std	7.110E+00	1.046E+01	1.132E+01	**2.974E+00**	2.189E+01
	Rank	2	3	**1**	4	5
$F_3(x)$	Minimum	**5.403E+01**	2.034E+02	6.042E+01	3.529E+02	1.000E+02
	Mean	**7.092E+01**	3.163E+02	1.296E+02	3.787E+02	2.956E+02
	Std	**7.166E+01**	1.160E+02	1.125E+02	8.554E+01	2.352E+02
	Rank	**1**	4	2	5	3
$F_4(x)$	Minimum	**2.330E+01**	4.187E+03	3.982E+01	1.315E+04	3.388E+04
	Mean	**4.462E+03**	3.274E+04	9.907E+03	1.827E+04	8.708E+04
	Std	**2.268E+04**	4.879E+04	4.366E+04	4.911E+04	6.250E+04
	Rank	**1**	4	2	3	5
$F_5(x)$	Minimum	**5.269E-03**	2.665E+00	7.890E-03	4.951E+01	1.186E-02
	Mean	4.081E+01	**3.687E+01**	1.184E+02	7.862E+01	1.501E+02
	Std	8.568E+01	1.139E+02	2.615E+02	**1.955E+01**	3.252E+02
	Rank	2	**1**	4	3	5
$F_6(x)$	Minimum	**5.277E+01**	6.707E+01	5.707E+01	3.287E+02	1.499E+02
	Mean	**7.622E+01**	2.918E+02	1.168E+02	4.427E+02	4.588E+03
	Std	**3.181E+02**	1.215E+03	4.753E+02	4.194E+02	4.892E+03
	Rank	**1**	3	2	4	5
$F_7(x)$	Minimum	**2.651E-08**	5.997E+02	1.385E-02	1.253E+03	9.981E-08
	Mean	**2.384E+02**	1.189E+03	6.532E+02	2.395E+03	2.782E+03
	Std	1.624E+03	3.605E+03	3.696E+03	**6.455E+02**	7.578E+03
	Rank	**1**	3	2	4	5
$F_8(x)$	Minimum	**4.903E-05**	1.728E+01	7.221E-03	3.256E+01	1.881E-03
	Mean	**3.226E+00**	2.701E+01	5.567E+00	3.919E+01	2.196E+01
	Std	1.084E+01	1.548E+01	1.854E+01	**3.387E+00**	3.489E+01
	Rank	**1**	4	2	5	3

Figures 5 to 12 depict the convergence trajectories of the seight test functions under a problem dimension of 30.

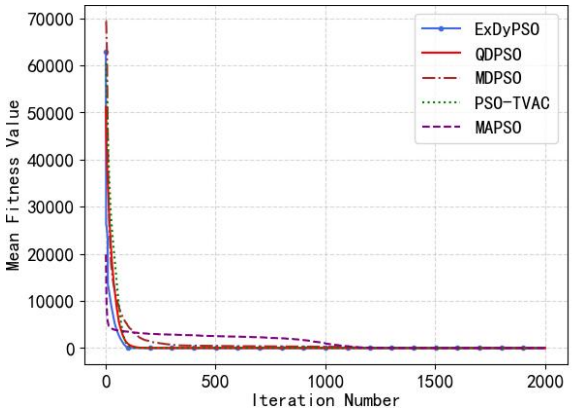

Fig. 5. Convergence performance on Sphere function F1(x).

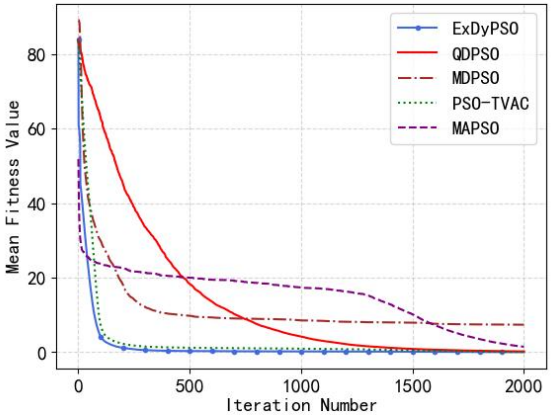

Fig. 6. Convergence performance on Sehwwefel P2. 21 function F2(x).

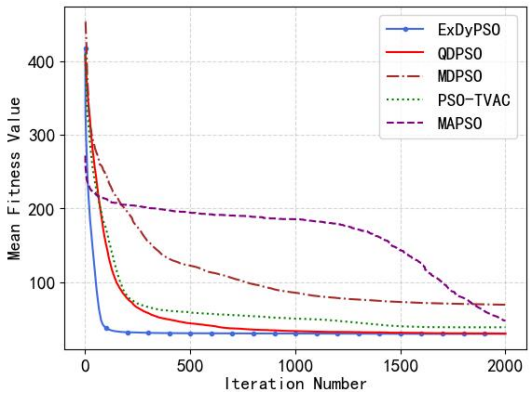

Fig. 7. Convergence performance on Rastrigin function F3(x).

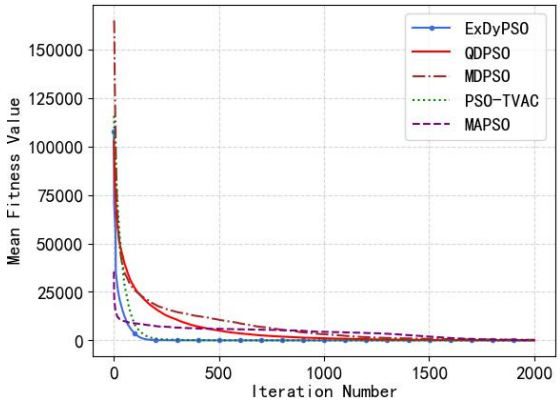

Fig. 8. Convergence performance on Schwefel P1. 2 function F4(x).

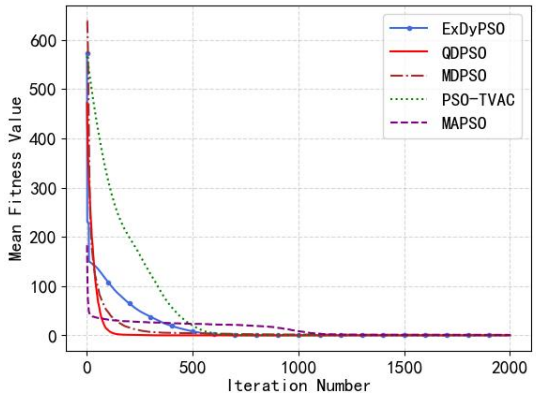

Fig. 9. Convergence performance on Griewank function F5(x).

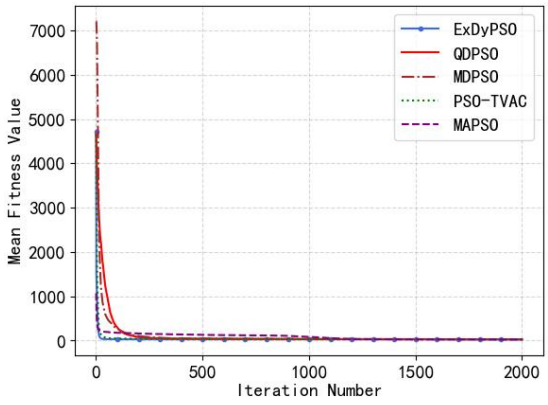

Fig. 10. Convergence performance on Rosenbrock function F6(x).

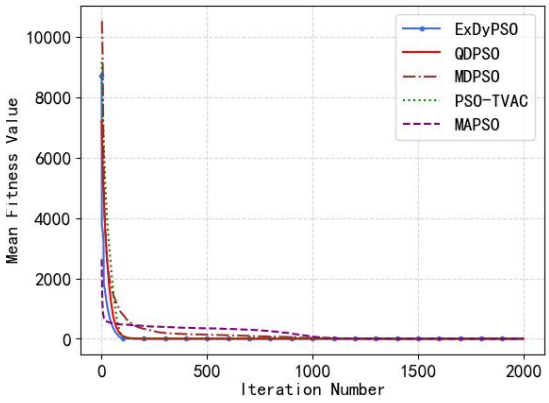

Fig. 11. Convergence performance on Sum Squares function F7(x).

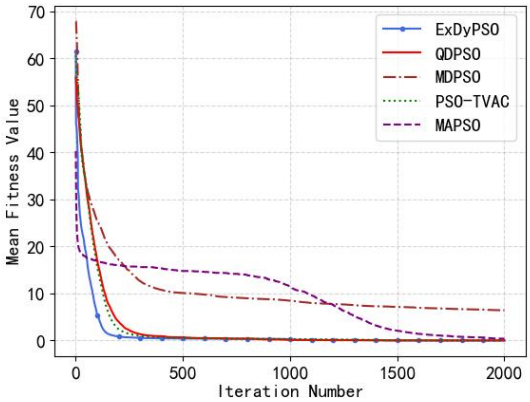

Fig. 12. Convergence performance on Alpine function F8(x).

It is apparent from all the figures that ExDyPSO showcases exceptional performance concerning convergence rate. In general, ExDyPSO exhibits highly stable performance. Furthermore, these figures depict ExDyPSO's adeptness at circumventing local optima when tackling intricate challenges, thereby enhancing convergence velocity.

V. CONCLUSION

In this paper, we presented a novel approach, Exponential Dynamic Inertia Weight for Particle Swarm Optimization (ExDyPSO), which uses exponentially distributed dynamic inertia weight improvement and dynamic acceleration factors improvement based on several iterations. These enhancements are geared towards preserving population diversity and optimizing the equilibrium between global and local search capabilities. Furthermore, we introduce a stochastic particle based jump-out strategy to facilitate particles' jump-out of the case of locally optimal solutions. We compared the efficacy of ExDyPSO against four state-of-the-art PSO algorithm variants. We also analyzed the population diversity by comparing with the classical PSO algorithm and the findings underscored the performance of our ExDyPSO algorithm, excelling in search accuracy and algorithmic robustness for both single-peak and multi-peak functions.

Our algorithm’s superior performance can be attributed to its utilization of dynamic parameters which afford it an extensive scope for population exploration, enhancing the likelihood of identifying optimal solutions. Additionally, ExDyPSO's jump-out strategy maximizes the utilization of information exchange among particles, and by combining the individual best position information from random particles,

new and better potential positions can be found in a wide range of regions. Nevertheless, further enhancement of the dynamic inertia weights is possible using an adaptive strategy that enables more effective adjustment to diverse circumstances during the search process.

ACKNOWLEDGMENT

This work was supported by the Natural Science Foundation of Inner Mongolia Autonomous Region 2021MS06003.

REFERENCES

[1] Kennedy, J. , Eberhart, R. : Particle swarm optimization. In: Proc. of ICNN'95, pp. 1942–1948. IEEE, Perth, WA, Australia (1995)

[2] Wang, D. , Zhao, S. , Chen, K. , Liu, S. : Parameter estimation of the classical fractal map based on a given Julia set's shape. Fractals 29(08), 2150247 (2021)

[3] Nasrabadi, A. M. , Moghimi, M. : Energy analysis and optimization of a biosensor-based microfluidic microbial fuel cell using both genetic algorithm and neural network PSO. Int. J. Hydrogen Energy 47(7), 4854–4867 (2022)

[4] Ansari, A. S. , et al. : Detection of Pancreatic Cancer in CT Scan Images Using PSO SVM and Image Processing. Biomed Res. Int. 2022, 1–7 (2022)

[5] Hammed, K. , Ghauri, S. A. , Qamar, M. S. : Biological inspired stochastic optimization technique (PSO) for DOA and amplitude estimation of antenna arrays signal processing in RADAR communication system. J. Sensors 2016, 1–10 (2016)

[6] Ratnaweera, A. , Halgamuge, S. K. , Watson, H. C. : Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans. Evol. Comput. 8(3), 240–255 (2004)

[7] Zhan, Z. H. , et al. : Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems. IEEE Trans. Cybern. 43(2), 445–463 (2013)

[8] Shi, Y. , Eberhart, R. C. : Empirical study of particle swarm optimization. In: Proc. of the 1999 Congress on Evolutionary Computation (CEC), pp. 1945–1950. IEEE, Washington, DC, USA (1999)

[9] Lin, A. , Sun, W. , Yu, H. , Wu, G. , Tang, H. : Global genetic learning particle swarm optimization with diversity enhancement by ring topology. Swarm Evol. Comput. 44, 571–583 (2019)

[10] Tao, X. , et al. : Self-Adaptive two roles hybrid learning strategies-based particle swarm optimization. Inform. Sci. 578, 457–481 (2021)

[11] Zhan, Z. H. , Zhang, J. , Li, Y. , Chung, H. S. H. : Adaptive particle swarm optimization. IEEE Trans. Syst. Man & Cybern. 39(6), 1362–1381 (2009)

[12] Chen, K. , et al. : A hybrid particle swarm optimizer with sine cosine acceleration coefficients. Inform. Sci. 422, 218–241 (2018)

[13] Tang, Y. , Wang, Z. , Fang, J. A. : Parameters identification of unknown delayed genetic regulatory networks by a switching particle swarm optimization algorithm. Expert Syst. Appl. 38(3), 2523–2535 (2011)

[14] Cheng, R. , Jin, Y. A Competitive Swarm Optimizer for Large Scale Optimization. IEEE Trans. Cybern. , 45(2), 191–204 (2015)

[15] Zeng, N. , et al. : A novel switching delayed PSO algorithm for estimating unknown parameters of lateral flow immunoassay, Cogn. Comput. 8(2), 143–152 (2016)

[16] Song, B. , Wang, Z. , Zou, L. : On global smooth path planning for mobile robots using a novel multimodal delayed PSO algorithm. Cogn. Comput. 9(1), 5–17 (2017)

[17] Thangaraj, R. , Pant, M. , Abraham, A. : A new diversity guided particle swarm optimization with mutation. In: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), pp. 294–299. IEEE, Coimbatore, India (2009)

[18] Wei, B. , et al. : Multiple adaptive strategies based particle swarm optimization algorithm. Swarm Evol. Comput. 57, 100731 (2020)

[19] Sun, J. , Feng, B. , Xu, W. : Particle swarm optimization with particles having quantum behavior. In: Proc. of the 2004 Congress on Evolutionary Computation, pp. 325–331. IEEE, Portland, OR, USA (2004)

[20] Meng, Zhenyu, et al. "PSO-sono: A novel PSO variant for single-objective numerical optimization. " Information Sciences 586 (2022): 176-191.

[21] Shukla, Amit K. , et al. "Engineering applications of artificial intelligence: A bibliometric analysis of 30 years (1988 – 2018). " Engineering Applications of Artificial Intelligence 85 (2019): 517-532.

[22] Camacho-Villalón, Christian L. , Marco Dorigo, and Thomas Stützle. "PSO-X: A component-based framework for the automatic design of particle swarm optimization algorithms. " IEEE Transactions on Evolutionary Computation 26. 3 (2021): 402-416.