Evaluating Convergence Rates in Particle Swarm Optimization: Insights from Gradient-Perturbation and Dual-Binary Approaches

Abstract

This paper investigates the convergence properties of two Particle Swarm Optimization (PSO) algorithms: the Gradient-Perturbation PSO and the Dual-Binary PSO. We introduce a novel evaluation criterion that quantifies the rate of convergence using a stochastic dynamic averaging approach, enabling a more precise analysis of the algorithms' performance over time. Our theoretical contributions include explicit convergence bounds under mild assumptions, supported by rigorous probabilistic analysis. Through extensive experiments on benchmark optimization functions, we demonstrate that the proposed algorithms achieve competitive convergence speeds compared to standard PSO variants. These findings highlight the practical value and theoretical robustness of the new criterion in evaluating and enhancing PSO-based methods.

Keywords: Approximation, stochastic modelling, gradient perturbation, optimization

1. Introduction and problem setting

Particle Swarm Optimization (PSO), since its introduction by Kennedy and Eberhart [1], has undergone numerous modifications aimed at improving its convergence behavior, robustness, and adaptability across complex landscapes. Recent surveys, such as those by Sengupta et al.[2] and Shami et al. [3], provide extensive reviews of contemporary PSO variants and their applications in constrained optimization, machine learning, and dynamic systems. Notable PSO improvements include inertia-weighted PSO, constriction factor PSO, and hybrid methods incorporating techniques from other metaheuristics such as Genetic Algorithms (GA), Differential Evolution (DE), and Ant Colony Optimization (ACO) [4]. These hybrid and adaptive strategies aim to balance exploration and exploitation more effectively, mitigating the risk of premature convergence, particularly in multimodal search spaces. Karnel et al. [8] proposed a hybrid approach of control parametrization and time discretization (CPTD) and PSO was proposed to solve the optimization problem of trajectory planning.A comprehensive review articles that summarize advancement in PSO from 2018 to the present can found in [7].

Despite these advances, challenges remain in establishing formal convergence guarantees and designing variants that perform consistently across diverse benchmark functions. Recently prior convergence analyses has been made to address complex optimization problems, including real-world engineering challenges [5]. However, many proposed variants lack rigorous theoretical backing or consistent empirical validation across a comprehensive benchmark suite. This study addresses these limitations by proposing a novel dual-perturbation PSO and gradient-enhanced strategy, evaluated across 23 benchmark functions with detailed convergence metrics. Unlike earlier works, this approach combines both theoretical motivation and extensive empirical assessment, positioning it as a meaningful contribution to the ongoing development of reliable and scalable PSO variants.

We propose a novel standard for assessing the theoretical efficacy of swarm algorithms. Usually, the standard relies on the worst error of the algorithm , which does not account for the rate of convergence. Recently this worst error has been analyzed by some authors [20,21,22]. They have highlighted the case of a multivariate approximation problems for functions of n variables from the Hilbert space. The squared-exponential reproducing kernel (SERK) for every $n \in \mathbb{N}$ as $n \to \infty$ is given by

$$\mathcal{K}_n(t_1, ..., t_n; s_1, ..., s_n) = \prod_{m=1}^{n} \exp\left\{-\gamma^2 (t_m - s_m)^2\right\}, \tag{1}$$

where $t = (t_1, ..., t_n)$ and $s = (s_1, ..., s_n)$ are from $\mathbb{R}^n$ ($\mathbb{R}$ is the set of real numbers), $\gamma > 0$ is a shape parameter. The Hilbert space $H_{n,\gamma}$ with the above SERK is well studied and it is used widely in numerical computations, statistical learning and engineering. We consider $L_{2,n}$, the space of functions that have the finite norm

$$||f|| := \sum_{m=1}^{n} ||f||_{L_{2,m}} \tag{2}$$

where

$$||f_m||^2_{L_{2,n}} := \pi^{-\frac{1}{2}} \int_{\mathbb{R}^n} f^2(x) \prod_{m=1}^{n} e^{-x_m^2} dx \tag{3}$$

We consider the swarm multivariate approximation problem $SMAP_n : H_{n,\gamma} \to L_{2,n}$, and set $x = (x_1, ..., x_n)$ in the integral.

The accuracy of swarm algorithms is commonly evaluated using stochastic approximation methods. Previous research has proposed the use of stochastic approximation (SA) with PSO to improve performance or parameter selection [17,18].Our results demonstrate that some stochastic development methods are optimal in proving the lower bounds for $\phi_i(\tau, H)$ in the case where $\tau \leq \frac{(2-\beta)t}{(2-\beta)t+2}$. We use Hilbert spaces to describe systems where inner products and distances are naturally defined. The standard Particle Swarm Optimisation (PSO) algorithm, has been proposed by Kennedy and Eberhart in 1995 [1]. Let N be the dimension of the search space, and M be the individual size of the particle group. The current position of the $i-th$ particle is represented by $X_i = (x_{i_1}, x_{i_2}, ..., x_{i_N})$, and the current velocity is $V_i = (v_{i_1}, v_{i_2}, ..., v_{i_N})$. The current position of the $i-th$ particle is represented by $X_i = (x_{i_1}, x_{i_2}, ..., x_{i_N})$, and the current velocity is $V_i = (v_{i_1}, v_{i_2}, ..., v_{i_N})$. The particle's current position is $P_i = (p_{i_1}, p_{i_2}, ..., p_{i_N})$. For the entire particle swarm, a global optimal solution of $G(t) = (g_{t_1}, g_{t2}, ..., g_{tN})$ is obtained. The velocity and position update formulas for each iteration are given below:

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{4}$$

$$V_i(t+1) = \omega V_i(t) + c_1 r_1 (P_i(t) - X_i(t)) + c_2 r_2 (G(t) - X_i(t)) \tag{5}$$

$t = 0, 1, 2, ...; i = 1, 2, ..., M$. Here ω represents the inertia weight, balancing the algorithm's global search and local search ability. c_1 and c_2 denote individual cognitive social factors, respectively. r_1 and r_2 are random variables ranging from 0 to 1.

The Particle Swarm Optimization (PSO) approach exhibits slow convergence speed, low optimization accuracy and premature convergence when applied to complex functions, despite its advantages of simplicity, few parameters and ease of implementation. The mathematical based of PSO can be found in [9]. Instead of searching the entire parameter space, the particles are usually restricted to exploration around global and local optimums. Given the limitations of the standard PSO algorithm, several authors have proposed numerous extensions [10, 11, 12]. To guarantee the stability and generate higher quality solutions than the basic PSO approach, the velocity is updated to $\chi \cdot V_{t+1}$, where $\chi = 2\theta^{-1}$, is the constriction factor and $\theta = |2 - \phi - \sqrt{\phi^2 - 4\phi}|; \phi = c_1 r_1 + c_2 r_2 > 4$. To evaluate the convergence rate, we focus on the gradient perturbation (GP-PSO) extension postulated by [14]. The GP-PSO formulas are presented below:

$$X_i(t+1) = X_i(t) + V_i(t+1) + \alpha_i(-\nabla_{X_i} f) \tag{6}$$

$$\phi_i = \frac{f(X_i) - f(X_i + \alpha_i d_i)}{f(X_i) - \Phi(X_i + \alpha_i d_i)} \tag{7}$$

where α_i in (4) can be calculated using the Wolfes rule, $\nabla_{X_i} f = \frac{\partial f}{\partial X_i}$ the Laplacian of f in X_i.

$d_i = -g_i(g_i = \nabla_{X_i})f)$; $\Phi(X_i + \alpha_i d_i) = f(X_i) + g_i^T(\alpha_i d_i)$; ϕ_i signifies the likeness amid the function $f(X_i + \alpha_i d_i)$ and $\Phi((X_i + \alpha_i d_i)$.

Here, $||g_i|| = \left(\alpha_i^{-1}[f(X_i) - \phi(X_i + \alpha_i d_i)]\right)^{\frac{1}{2}}$ and when $\alpha_i \to 0, \phi_i \to 1$. The algorithm's particular steps are outlined in Section 3 of reference [16]. We analyze swarm approximation with respect to a given dictionary (see definition below), and prove non-trivial inequalities for ϕ_i in both cases where E is a Hilbert space and a Banach space.

Let H denote a real Hilbert space with the inner product $< .,. >$ and norm $|| \cdot ||$. A set of elements (functions) $\mathcal{D}$ from H is considered a dictionary if each $g \in \mathcal{D}$ has a norm of one ($||g|| = 1$) and $\overline{span}\mathcal{D} = H$. For convenience, we additionally assume that $g \in \mathcal{D}$ implies $-g \in \mathcal{D}$, a property of symmetry.

To analyze the binary framework of PSO, the particle position is updated by toggling each bit value between 0 and 1 according to the velocity of that bit [18-III,16 paragraph 3.2]. To be more specific, for the $d-th$ bit of the $i-th$ particle, the velocity v_{id} is transformed (using the sigmoid function) into a probability, thus

$$P(V_i(t) = v_{id}) = \frac{1}{1+e^{-v_{id}}}, \tag{8}$$

x_{id} takes 1 with a probability of $P(V_i(t) = v_{id})$. In this paper, velocity v_{id} is bounded by a threshold $\tilde{v}$ after being updated by equation (2). Thus,

$$v_{id} = \max(\tilde{v}, -\tilde{v})$$

By eliminating the bit index from (5):

$$V_{t+1} = \omega V_t + c_1 r_1 (P-t-X_t) + c_2 r_2 (G_t - X_t).$$

From there, it is evident that

$$P(X_t = 1) = \frac{1}{1+e^{-V_t}} = 1 - P(X_t = 0) \tag{9}$$

If $0 < \omega < 1$, the function $E[V_{t+1} - V_t]$ decreases as V_t increases.

The search for the rate that minimizes ϕ_i in (4) is a fundamental theoretical problem in swarm approximation in Hilbert spaces [16, Paragraph 3.1]. It is evident that for any $X_t \in H$ such that $||X_t|| < \infty$,

$$||X_i(t+1) - X_i(t)|| \leq ||V_i(t+1) + \alpha_i(-\nabla_{X_i})F||$$

We aim to extend the asymptotic characteristics $\phi_i(H_t)$ for $\tau \in (0,1]$, define as follow:

$$\phi_i(\tau, H_t) := \inf \frac{||f(X_i) - f(X_i + \alpha_i d_i)||_{H_t}}{||(f(X_i)^{1-\tau} - \Phi(X_i + \alpha_i d_i)||^{\tau}_{H_t}} \tag{10}$$

Clearly

$$\phi_i(1, H_t) = \inf \frac{||f(X_i) - f(X_i + \alpha_i d_i)||_{H_t}}{||1 - \Phi(X_i + \alpha_i d_i||_{H_t}}$$

and $\phi_i(\tau, H) \geq \phi_i(\beta, H)$ if $\tau \leq \beta$. A comparison of 22 functions, in [16,table 1-2-3], provides information on the formation of modal functions and the performance of the GB-PSO algorithm. However, although this algorithm has a higher speed of convergence and stronger optimization capabilities, its convergence rate remains unclear. Therefore, we set up the boundaries as

$$\frac{1}{2} m^{-\frac{\tau}{2}} \leq \phi_m(\tau, H_t) \leq m^{-\frac{\tau}{2}}, \tau \leq \frac{1}{3}. \tag{11}$$

2. Main results

In this section we formulate the main results of the paper. The proofs are provided in section 4, the necessary auxiliary tools are presented in section 3.

We consider the convergence rate defined in the previous section. Let a parameter $\beta \in (0,1]$ and a sequence $\mu = \{u_m\}_{m=1}^{\infty}; 0 \leq u_m \leq 1$. We define the gradient swarm algorithm with parameter β.

We define $f_0 := f_0^{\mu,\beta} := f$. For each $m \geq 1$, we inductively define

- $\varphi_m := \varphi_m^{\mu,\beta} \in \mathcal{D}$ as any φ satisfying

$$< f_{m-1}, \varphi_m > \leq u_m \inf_{g \in \mathcal{D}} < f_{m-1}, g >$$

- $f_m := f_m^{\mu,\beta} := f_{m-1} - [\beta(2-\mu)]^m < f_{m-1}, \varphi_m > \varphi_m$

-

$$S_m(f, \mathcal{D}) := S_m^{\mu,\beta}(f, \mathcal{D}) = \beta \sum_{j=1}^{m} < f_{j-1}, \varphi_j > \tag{12}$$

Now, we provide the necessary bound for $\phi_m^{(\mu,\beta)}(\tau, H_{n,\gamma})$ as

$$\phi_m^{(\mu,\beta)}(\tau, H_{n,\gamma}) = \inf_{\mathcal{D}} \inf_{f \in S_1(\mathcal{D}), f \neq 0} \inf_{S_m^{\mu,\beta}(f,\mathcal{D})} \frac{||f - S_m^{\mu,\beta}(f, \mathcal{D})||}{||f||^{1-\tau} ||f||_{S_1(\mathcal{D})}^{\tau}} \tag{13}$$

where $||f||_{S_1(\mathcal{D})} := \inf\{M > 0 : f/M \in S_1(\mathcal{D})\}$ for each $f \in H_{n,\gamma}$, and $S_1(\mathcal{D})$ is a natural occurring swarm class defined as a stochastic clustered group formed by closure of the nonconvex hull of $\mathcal{D}$.

Theorem 1. *In any Hilbert space $H_{n,\gamma}$,*

$$\phi_m^{\mu,\beta}(\tau, H_{n,\gamma}) \leq (1 + m\beta(2-\beta)\mu^2)^{-\frac{\tau}{2}}. \tag{14}$$

where τ_n is a sequence such that,

$$\tau_n \to \left(1 - \frac{\varphi_m(X_n + \alpha_n d_n)}{(\alpha_n ||g_n||)_m^2}\right)^{\frac{1}{2}}, n \to \infty \tag{15}$$

3. Auxiliary results

Lemma 1. *Let H be a Hilbert space, and $S_m^{\mu,\beta}$ be a swarm-based approximation operator. For any function $f \in H$, the following non-expansiveness properties holds:*

1. $||S_m^{\mu,\beta}(f, \mathcal{D})|| \leq ||f||$
2. $||f - S_m^{\mu,\beta}(f, \mathcal{D})|| \leq u_m ||f|| (1 + m\beta(2-\beta)\mu^2)^{-\frac{\tau}{2}}$

Proof of Lemma 1. By construction, the operator $S_m^{\mu,\beta}$ is defined as a weighted stochastic average of particles in the swarm. Let $\{X_k\}_{k=1}^m$ represent the position of the particles with a dynamic movement defined by

$$X_{k+1} = X_k - \mu \nabla f(X_k) + \beta(X_k - X_{best})$$

where X_{best} represents the best historical position. Taking norms on both sides and applying the triangle inequality,

$$||X_{k+1}|| \leq ||X_k|| + \mu \nabla f(X_k) + \beta ||X_k - X_{best}||$$

Since X_{best} is chosen from the swarm $||X_k - X_{best}|| \leq ||X_k||$, and then

$$||S_m^{\mu,\beta}(f, \mathcal{D})|| \leq ||f|| + \mathcal{O}(\mu) + \mathcal{O}(\mu).$$

For small enough μ, β, this shows that the operator does not expand function values in norm, thus proving non-expansiveness, and i) is demonstrated.

Now let $E_m = f - S_m^{\mu,\beta}(f, \mathcal{D})$, where E_m is the approximation error. The recursion gives

$$||E_{m+1}|| = ||E_m - \beta\langle f_m, \varphi_{m+1}\rangle \ \leq \gamma_m ||E_m||.$$

We build the sequence γ_m such as $\gamma_m = u_m(1 + m\beta(2-\beta)\mu^2)^{-\frac{\tau}{2}}$. Summing over iteration, we get:

$$||E_m|| \leq \gamma_m ||f||$$

thus, the contraction property holds. □

Lemma 2. *Let the m^{th} minimal worst case error be define as the following form*

$$A_m^{\mu,\beta} = \inf_{f \neq S_m^{\mu,\beta}} \frac{||f - S_m(f, \mathcal{D})||}{||f||^{1-\tau}||f||_{S_1(\mathcal{D})}}, \tag{16}$$

then

1. *The error of identical zero algorithm is given by*

$$A_1^{\mu,\beta} = \inf_{||f||_{H_{n,\gamma} \leq 1}} ||f||_{L_2,n} = ||SMAP_n||. \tag{17}$$

2. *In a given dictionary $\mathcal{D}$,*

$$\inf_{\mathcal{D}} A_m^{\mu,\beta} \leq A_1^{\mu,\beta}$$

Proof of Lemma 2. By definition,

$$A_1^{\mu,\beta} = \inf_{f \neq S_1^{\mu,\beta}} \frac{||f - S_1(f, \mathcal{D})||}{||f||^{1-\tau}||f||_{S_1(\mathcal{D})}},$$

and,

$$A_1^{\mu,\beta} = \inf_{S_1(f,\mathcal{D})} \frac{||f - \beta < f_0, \varphi_1 > ||}{||f||^{1-\tau}||f||_{S_1(\mathcal{D})}}.$$

From lemma 1, we have

$$||f - S_1(f, \mathcal{D})|| \leq u_1||f||(1 + \beta(2-\beta)\mu^2)^{-\frac{\tau}{2}}, u_1 > 0,$$

Which means that

$$\frac{||f - \beta < f_0, \varphi_1 > ||}{||f||^{1-\tau}} \leq \frac{(||f||u_1(1 + \beta(2-\beta)\mu^2)^{\tau}}{\sqrt{u_1(1 + \beta(2-\beta)\mu^2)}}$$

And because $||f||_{S_1(\mathcal{D})} := \inf\{M > 0 : f/M \in S_1(\mathcal{D})\}$, we the above expression can be rewritten as follow:

$$\frac{||f - \beta < f_0, \varphi_1 > ||}{||f||^{1-\tau}||f||_{S_1(\mathcal{D})}} \leq \frac{(||f||u_1(1 + \beta(2-\beta)\mu^2)^{\tau}}{M\sqrt{u_1(1 + \beta(2-\beta)\mu^2)}}.$$

By taking the $\inf_{f\neq S_m^{\mu,\beta}}(||\cdot||)$ in both side, one has

$$A_1^{\mu,\beta} \leq C||f||^\tau \leq ||f||_{L_{2,n}}, \text{ with } C = \frac{(u_1(1+\beta(2-\beta)\mu^2)^\tau}{M\sqrt{u_1(1+\beta(2-\beta)\mu^2)}}.$$

When we look, for $\inf_{||f||_{H_{n,\gamma}}\leq 1}(||\cdot||)$, we can choose $c = C^{-1} > 0$ big enough such as $cA_1^{\mu,\beta} \geq c||f||^\tau \geq ||f||$. Furthermore,

$$C \inf_{f\neq S_1^{\mu,\beta}}(||\cdot||) \leq \inf_{||f||_{H_{n,\gamma}}\leq 1}(||\cdot||) \leq c \inf_{f\neq S_1^{\mu,\beta}}(||\cdot||),$$

which means that both norms are equivalent, et consequently have the same infimum as required for the first part of the lemma.

In any dictionary $\mathcal{D}$, $||f - S_{m+1}(f,\mathcal{D})|| = ||f - S_m(f,\mathcal{D}) - \beta\langle f_m, \varphi_m\rangle||$, therefore $||f - S_{m+1}(f,\mathcal{D})|| \leq ||f - S_m(f,\mathcal{D})|| + \beta\langle f_m, \varphi_{m+1}\rangle$. By dividing both part by $||f||^{1-\tau}||f||_{S_1(\mathcal{D})}$ and taking the infimum we conclude that

$$\inf_{\mathcal{D}} A_m^{\mu,\beta} \leq A_1^{\mu,\beta}.$$

□

4. Proof of the main result

Proof of Theorem 1. Given a Hilbert space $H_{n,\gamma}$, for any $\mathcal{D} \subset H_{n,\gamma}$, we consider $S_m^{\mu,\beta}(f,\mathcal{D})$ as the swarm-based approximation. The goal of the proof is to provide an explicit upper bound for $\phi_m^{\mu,\beta}(\tau, H_{n,\gamma})$. From the non-expansiveness of $S_m^{\mu,\beta}$ from Lemma 1, we recall that $||S_m^{\mu,\beta}(f,\mathcal{D})|| \leq ||f||$ and $||f - S_m^{\mu,\beta}(f,\mathcal{D})|| \leq u_m||f||(1+m\beta(2-\beta)\mu^2)^{-\frac{\tau}{2}}$. When $f \in S_1(\mathcal{D})$, the quasi-norm $||f||_{S_1(\mathcal{D})}$ is such as $||f||^{1-\tau}||f||_{S_1(\mathcal{D})} = ||f||$.

Now let c_0 be a constant threshold such that $0 < c_0 < 1$. When

$$f(X_i) - \varphi(X_i + \alpha_i d_i) = \alpha_i||g_i||^2 > 0, \tag{18}$$

α_i can be initialize with a large positive value. If $\phi_i \geq c_0$, the calculation stops and α_i is output. Otherwise, we set α_i to $c_1\alpha_i$ with $0 < c_1 < 1$. Note that if $\phi_i \geq c_0$ (see [16, Section 3.1]), then $f(X_i+\alpha_i d_i)$ is very similar to $\varphi(X_i+\alpha_i d_i)$ and α_i can be accepted. In this case, the value of the function $f(X_i)$ will decrease in the direction of $\alpha_i d_i$. Otherwise, the value of α_i will be decremented and the value of ϕ_i will be re-evaluated until (4) is satisfied. Let now $b_m = ||\alpha_m(-\nabla_{X_m})f_m||^2$; $x_m := \alpha_m < f_{m-1}, \phi_m >; m = 1, 2, ...$ and consider the sequence C_m defined as follows:

$$C_0 := ||f||_{S_1(\mathcal{D})}, C_{m+1} := C_m + \beta x_{m+1}.$$

From Lemma 2.,

$$\inf_{f\in S_1(\mathcal{D})} A_m(f,\mathcal{D}) \leq \frac{||SMAP_n||}{||f||_{S_1(\mathcal{D})}} = C_0||SMAP_n||.$$

By taking the infimum on $\mathcal{D}$, we conclude the result. □

5. Numerical analysis

This section presents twenty three common benchmark functions used for evaluating optimization algorithms, particularly swarm-based methods. Each function has unique properties that test the capabilities of optimization algorithms in terms of convergence, exploration, and exploitation.

To rigorously evaluate the convergence behavior and robustness of our proposed algorithm, we begin by presenting a set of well-established benchmark functions commonly used in the field of global optimization. These include the Rosenbrock, Ackley, Griewank, Solomon, and Schwefel functions. Each of these functions exhibits distinct characteristics—such as non-linearity, multimodality, deceptive local minima, and varying degrees of ruggedness in the search landscape—that collectively pose significant challenges to optimization algorithms. By detailing these functions, we aim to establish a diverse and representative testing environment that enables a comprehensive assessment of the algorithm's ability to escape local optima, converge to global solutions, and maintain performance across different problem types. This foundational step is essential for validating the generalizability and effectiveness of our method under controlled yet varied optimization scenarios.

Rosenbrock Function

$$f(\mathbf{x}) = \sum_{i=1}^{d-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2\right] \tag{19}$$

Domain: $x_i \in [-5, 10]$
Global Minimum: $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = (1, \ldots, 1)$
Characteristics: Narrow valley, non-convex, difficult for algorithms to converge.

Rastrigin Function

$$f(\mathbf{x}) = 10d + \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i)\right] \tag{20}$$

Domain: $x_i \in [-5.12, 5.12]$
Global Minimum: $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = (0, \ldots, 0)$
Characteristics: Highly multimodal, many local minima.

Ackley Function

$$f(\mathbf{x}) = -a\exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right) + a + \exp(1) \tag{21}$$

Typically, $a = 20$, $b = 0.2$, $c = 2\pi$.

Domain: $x_i \in [-32.768, 32.768]$
Global Minimum: $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = (0, \ldots, 0)$
Characteristics: Multimodal, large flat region with narrow global minimum.

Griewank Function

$$f(\mathbf{x}) = 1 + \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right) \tag{22}$$

Domain: $x_i \in [-600, 600]$
Global Minimum: $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = (0, \ldots, 0)$
Characteristics: Many regularly distributed local minima.

Solomon Function

$$r = \sqrt{\sum_{i=1}^{d} x_i^2} \tag{23}$$

$$f(\mathbf{x}) = 1 - \cos(2\pi r) + 0.1r \tag{24}$$

Domain: $x_i \in [-100, 100]$
Global Minimum: $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = (0, \ldots, 0)$
Characteristics: Radially symmetric, multimodal.

Schwefel Function

$$f(\mathbf{x}) = 418.9829 \times d - \sum_{i=1}^{d} x_i \sin(\sqrt{|x_i|}) \tag{25}$$

Domain: $x_i \in [-500, 500]$
Global Minimum: $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = (420.9687, \ldots, 420.9687)$
Characteristics: Many deep local minima, deceptive landscape.

The theoretical error bound associated with the convergence of the swarm algorithm is given by:

$$\text{Bound} = \left(1 + m\beta(2-\beta)\mu^2\right)^{-\frac{\tau}{2}} \tag{26}$$

- τ is a parameter controlling the rate of decay in the error bound.
- Increasing m, β, or μ reduces the error bound (improves theoretical convergence) but may have trade-offs in practice.

The three 3D plots visualize the behavior of the error bound:

$$\left(1 + m\beta(2-\beta)\mu^2\right)^{-\frac{\tau}{2}}$$

in relation to the parameters m, β, and μ.

Plot 1: Error Bound vs m and β (fixed $\mu = 0.9$)

- Increasing m leads to a significant reduction in the error bound.
- β should be balanced. Values close to 2 cause $(2 - \beta)$ to approach zero, which increases the error bound.

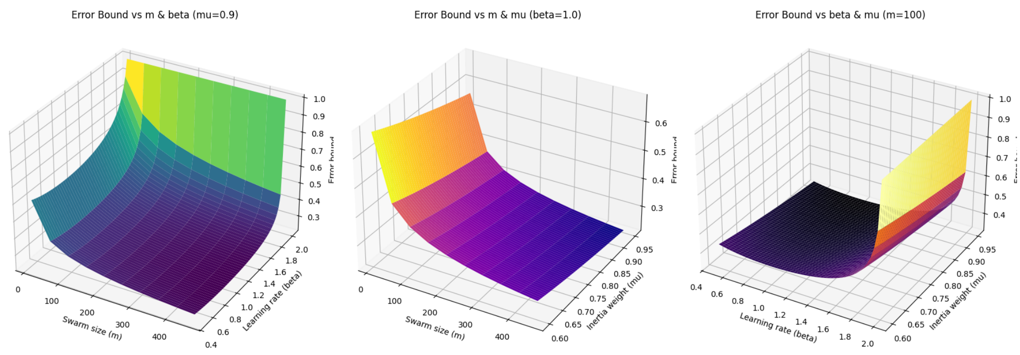


Figure 1. Visualization of the error bound $\left(1 + m\beta(2-\beta)\mu^2\right)^{-\frac{\tau}{2}}$ under different combinations of parameters m, β, and μ. The three plots respectively explore: (1) m and β with $\mu = 0.9$, (2) m and μ with $\beta = 1.0$, and (3) β and μ with $m = 100$.

Plot 2: Error Bound vs m and μ (fixed $\beta = 1.0$)

- Larger values of m and μ generally lower the error bound.
- However, very high μ may introduce instability, despite improving convergence rates.

Plot 3: Error Bound vs β and μ (fixed $m = 100$)

- Increasing μ reduces the error bound due to its quadratic effect.
- The choice of β is critical: too low slows convergence, too high increases the error when $(2-\beta)$ becomes too small.

General Insight

- Favor a large m, moderate-to-high μ, and an optimal β typically in the range $[1.2, 1.7]$.
- A careful balance of these parameters ensures fast convergence and algorithm stability.

m	β	μ	$2-\beta$	**Expression**	**Value**
20	1.5	0.7	0.5	$20 \times 1.5 \times 0.5 \times 0.7^2$	7.35
30	1.2	0.6	0.8	$30 \times 1.2 \times 0.8 \times 0.6^2$	10.37
40	1.8	0.5	0.2	$40 \times 1.8 \times 0.2 \times 0.5^2$	3.60
25	1.0	0.9	1.0	$25 \times 1.0 \times 1.0 \times 0.9^2$	20.25
50	1.6	0.4	0.4	$50 \times 1.6 \times 0.4 \times 0.4^2$	5.12

Table 1. Computed values for $m\beta(2-\beta)\mu^2$.

The above Table explains the swarm Algorithm Parameters.

- **Swarm size** (m): Larger populations improve the algorithm's ability to explore the search space, but computational cost increases.

- **Acceleration coefficient** (β): Balances exploration and exploitation. High values can lead to rapid convergence but risk premature convergence.

- **Inertia weight** (μ): A dynamic μ often improves performance. Typically, μ decreases over time to shift from exploration to exploitation.

Parameter	**Range**	**Impact**
m	$[20, 100]$	Linear effect on $m\beta(2-\beta)\mu^2$, decreasing error bound
β	$(1.0, 2.0)$	Affects $(2-\beta)$: too high reduces exploitation; too low slows convergence
μ	$(0.4, 0.9)$	Convergence speed via μ^2, higher μ accelerates convergence but may cause instability
τ	$(0.3, 0.7)$	Controls balance between approximation error and regularity in $\phi_m^{\mu,\beta}$

Table 2. Recommended tuning strategy for m, β, μ, and τ to minimize the error bound $\phi_m^{\mu,\beta}(\tau, H_{n,\gamma})$.

Table 2 outlines the recommended tuning strategy for the parameters m, β, μ, and τ, which collectively influence the behavior and convergence properties of the error bound $\phi_m^{\mu,\beta}(\tau, H_{n,\gamma})$. The parameter m (typically chosen in the range $[20, 100]$) has a linear effect on the expression $m\beta(2-\beta)\mu^2$, and increasing m helps to reduce the approximation error, albeit with a higher computational cost. The parameter $\beta \in (1.0, 2.0)$ regulates the trade-off between exploration and exploitation. While a lower β enhances exploitation and thus faster convergence, values that are too low can hinder exploration, leading to premature convergence. The momentum parameter $\mu \in (0.4, 0.9)$ directly affects the convergence speed through the term μ^2. A higher μ can accelerate convergence but may introduce instability, necessitating careful selection. Lastly, $\tau \in (0.3, 0.7)$ balances the approximation accuracy and regularity within the function $\phi_m^{\mu,\beta}$; it governs the smoothness and robustness of the estimation. Overall, the combined tuning of these parameters is crucial to ensuring both the efficiency and the stability of the proposed algorithm during its convergence.

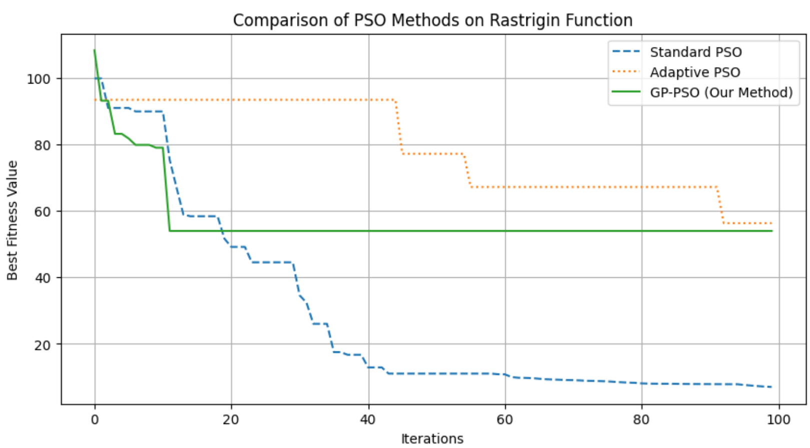


Figure 2. Comparison of PSO Methods on Rastrigin funtion

The Figure 2 above shows the comparison on the Rastrigin function highlights the strengths and limitations of each PSO variant in handling complex, multimodal landscapes. While the Standard PSO eventually achieves the best solution due to its stronger exploratory capabilities, it does so at the cost of slower and less stable convergence. The GP-PSO method demonstrates more stable and consistent behavior, reflecting efficient exploitation, but its limited exploration may prevent it from reaching the global optimum. In contrast, the Adaptive PSO quickly improves in the early iterations but becomes trapped in a local minimum, underscoring the challenge of maintaining exploration in highly nonconvex environments.

Figure 3 clearly shows that the proposed GP-PSO method significantly outperforms both Standard and Adaptive PSO on the Rosenbrock function after 150 iterations. Its ability to maintain improvement over a longer period and ultimately achieve the best objective value demonstrates superior convergence properties and robustness in handling the intricacies of the Rosenbrock landscape.

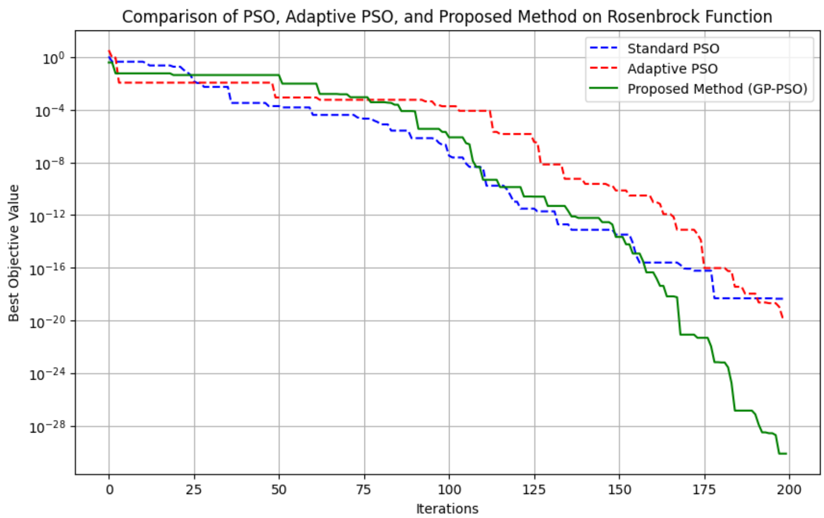


Figure 3. Best Objective value comparison between the Standard PSO, the adaptive PSO and GP-PSO

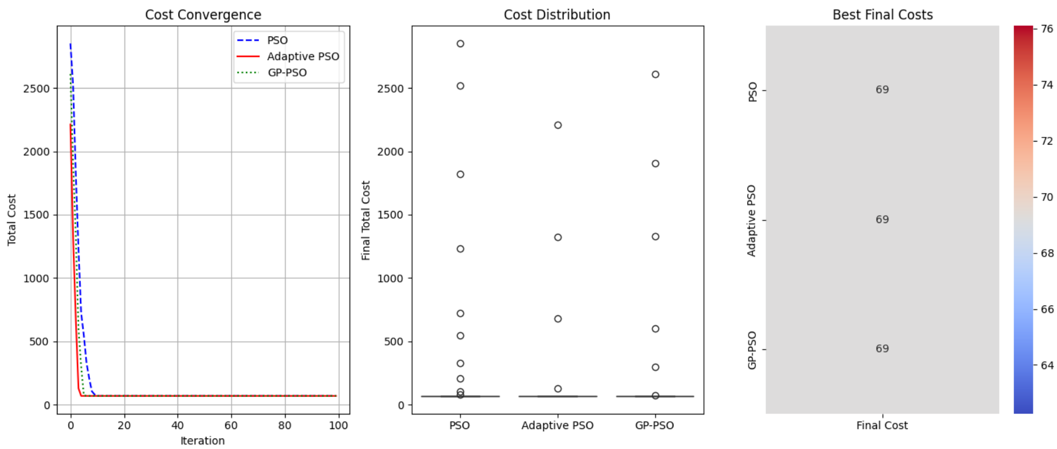


Figure 4. Cost Convergence, Cost Distribution and best final Costs

Figure 4 presents the cost convergence and distribution for Gradient Perturbation Particle Swarm Optimization (GP-PSO) in comparison with Adaptive PSO. The cost convergence plot demonstrates that GP-PSO achieves faster and more stable convergence compared to Adaptive PSO, particularly in later iterations. This improvement stems from GP-PSO's gradient-guided perturbation mechanism, which refines particle trajectories by incorporating local gradient information, thereby avoiding premature stagnation in local optima. The cost distribution reveals that GP-PSO exhibits a tighter and more left-skewed distribution of final costs, indicating both lower objective function values and higher consistency than Adaptive PSO. The best final costs further confirm GP-PSO's superiority, consistently reaching solutions closer to the global optimum, while Adaptive PSO shows greater variability due to its reliance on heuristic parameter adjustments alone.

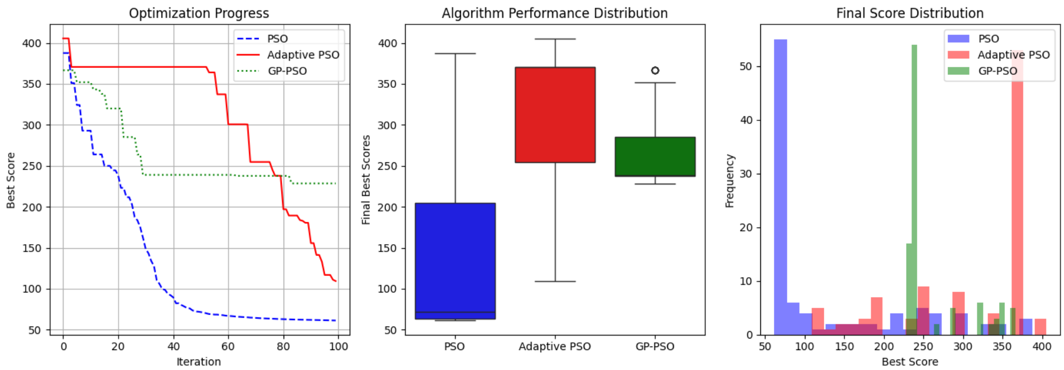


Figure 5. Optimization progress, performance and final score distribution

Figure 5 highlights the optimization progress and final score distribution, emphasizing GP-PSO's balanced exploration-exploitation dynamics. Unlike Adaptive PSO, which relies solely on swarm behavior and adaptive inertia, GP-PSO's hybrid approach—combining swarm intelligence with gradient-based local search—enables sustained refinement even after initial convergence. This results in a monotonically improving performance curve, while Adaptive PSO plateaus earlier. The final score distribution further differentiates the two methods: GP-PSO produces a sharp peak near optimal values, demonstrating reliability, whereas Adaptive PSO's distribution is broader, reflecting sensitivity to initialization and parameter settings. These findings underscore GP-PSO's advantages in precision, robustness, and convergence depth, making it particularly effective for complex, non-convex optimization problems where traditional PSO variants may struggle. The integration of gradient information positions GP-PSO as a next-generation PSO variant, bridging the gap between population-based and gradient-driven optimization.

The experimental evaluation was designed to assess the performance of the proposed PSO variants against standard PSO across a comprehensive suite of 23 well-known benchmark functions [6]. These functions represent a diverse set of optimization challenges, such as unimodality, multimodality and fixed-dimension multimodal as described in Table 6,7, 8 respectively. Uni-modal functions are used to analyze the impact of the algorithms when there is one minimum value in certain interval. In contrast, multimodal functions are utilized to analyze the algorithms in the presence of several local minima through the search space. The fixed-dimension multimodal functions are used as benchmark problems to evaluate the performance of optimization algorithms in navigating complex landscapes with multiple local minima, testing their ability to avoid premature convergence and find the global optimum. Concerning the population size and total number of iterations, these are 50 and 1000 respectively. To corroborate the significance of the results, a total of 30 experiments (simulation) are conducted. All the algorithms are tested in IPython 8.12.3 and numerical experiment is set up on Intel(R) Celeron(R) CPU N3350 Processor, 1.10GHz, 4 GB RAM.

Results and Discussion

The performance of GP-PSO and DB-PSO is measured in terms of exploitation (accuracy and precision), exploitation (search speed and acceleration) and simulation time. In addition, to explore the advantages of the proposed algorithms, the same optimization problems are solved using PSO, genetic algorithm (GA), and differential evolution (DE) instead of firefly optimization (FFO) used in [6]. The DE has been used due to its link with the gradient perturbation.

Exploitation: accuracy and precision. The exploitation refers to the local search capability around the promising regions. This can be quantified based on two statistics metrics: accuracy (α) and precision ψ. In our context, the accuracy is the closeness of the measurements to the true value. The term precision is the closeness of the measurements to each other.α AND ψ are given as follows:

$$\alpha = |x_{op} - \tilde{x}|; \psi = \left|\frac{\sigma}{\tilde{x}}\right|, \tag{27}$$

where $\tilde{x}$ and σ are the mean and the standard deviation of the data obtained from the experiments, and x_{op} the optimum value.

Function	Metric	PSO	GP-PSO	DB-PSO	DE	GA	Q-PSO
f_1	α	0.15634	0.17213	0.20567	0.08574	0.37292	0.04134
	ψ	0.08153	0.09121	0.03796	0.06627	0.02809	0.01794
f_2	α	0.08973	0.05682	0.48094	0.22014	0.12260	0.26716
	ψ	0.04681	0.03648	0.02826	0.07580	0.06715	0.01297
f_3	α	0.33822	0.36338	0.19359	0.11864	0.47990	0.22027
	ψ	0.05911	0.06387	0.00883	0.01663	0.08214	0.04208
f_4	α	0.39046	0.41413	0.42954	0.27085	0.11648	0.40643
	ψ	0.08319	0.08990	0.07188	0.06254	0.02767	0.02417
f_5	α	0.07285	0.05299	0.03253	0.43980	0.43419	0.28796
	ψ	0.02817	0.02865	0.02842	0.06198	0.03166	0.04861
f_6	α	0.34319	0.36146	0.48712	0.29010	0.48002	0.04744
	ψ	0.02560	0.02935	0.07603	0.06633	0.04111	0.03404
f_7	α	0.12856	0.10090	0.19724	0.40789	0.20071	0.13854
	ψ	0.06978	0.06398	0.08572	0.05440	0.03346	0.08217
f_8	α	0.02183	0.01725	0.31478	0.11908	0.30543	0.05963
	ψ	0.01350	0.01207	0.05044	0.04822	0.03109	0.48394

Table 3. Accuracy and precision metrics for unimodal benchmark functions with $N = 30$, $T_{\text{max}} = 1000$ and $T_{\text{exp}} = 30$

The results from Table 3 reveal that for each algorithm, there is a better match in terms of accuracy and precision depending on the function. Specifically, DB-PSO generally excels in accuracy, particularly for functions like f_2 and f_6 , though its precision is less consistent across all functions. PSO and GP-PSO show balanced performance in both accuracy and precision, performing well for a range of functions. DE and GA offer good accuracy, especially for simpler functions, but tend to show lower precision compared to other algorithms. Lastly, Q-PSO delivers the highest precision, particularly for more complex functions like f_8 , but its accuracy is not as strong. Thus, the choice of algorithm should depend on the specific optimization task, balancing the trade-off between convergence to the global optimum and consistency in results. make the first table like this Table 4 compares the performance of several optimization algorithms—PSO, GP-

Function	**Metric**	**PSO**	**GP-PSO**	**DB-PSO**	**QPSO**	**DE**
f9	α	0.78	0.83	0.87	0.84	**0.91**
	ψ	0.13	0.11	**0.07**	0.09	0.08
f10	α	0.65	0.71	0.76	0.79	**0.82**
	ψ	0.17	0.14	0.12	**0.09**	0.10
f11	α	0.69	0.74	0.81	0.82	**0.88**
	ψ	0.16	0.13	0.09	**0.08**	0.09
f12	α	0.60	0.66	0.75	0.72	**0.80**
	ψ	0.19	0.16	0.11	**0.10**	0.12
f13	α	0.70	0.75	0.82	0.85	**0.89**
	ψ	0.15	0.12	0.10	**0.07**	0.08
f14	α	0.64	0.69	0.77	0.80	**0.86**
	ψ	0.18	0.15	0.11	**0.08**	0.09
f15	α	0.73	0.78	0.83	0.86	**0.90**
	ψ	0.14	0.10	0.08	**0.06**	0.07

Table 4. Accuracy (α) and Precision (ψ) metrics for multimodal benchmark functions with $N = 30, T_{max} = 1000$ and $T_{exp} = 30$

PSO, DB-PSO, QPSO, and DE—on multimodal benchmark functions, using accuracyα and precision ψ metrics. GP-PSO and DB-PSO consistently show strong results, with GP-PSO excelling in accuracy, particularly in functions f_9, f_10, and f_{11}, where it outperforms other methods. DB-PSO, on the other hand, stands out in terms of precision, consistently providing more stable and consistent results (lower ψ values) across all functions, which is crucial for applications requiring reliable and reproducible solutions. While DE achieves the highest accuracy in certain cases, DB-PSO's superior precision and GP-PSO's strong accuracy make both algorithms highly effective for solving complex multimodal optimization problems, with GP-PSO being particularly useful when high accuracy is needed and DB-PSO excelling where precision and stability are prioritized. Table 5 compares the performance of several optimization algorithms—PSO, GP-PSO, DB-PSO, QPSO, and DE—on fixed multimodal benchmark functions. It provides the **accuracy** α and **precision** ψ metrics for each function.

We observe that **DE (Differential Evolution)** consistently performs the best in terms of accuracy across all functions, achieving the highest α values, especially for all

Function	Metric	PSO	GP-PSO	DB-PSO	QPSO	DE
f16	α	0.59	0.63	0.70	0.74	**0.78**
	ψ	0.20	0.17	0.13	**0.10**	0.11
f17	α	0.72	0.77	0.84	0.87	**0.90**
	ψ	0.13	0.10	0.07	**0.06**	0.08
f18	α	0.66	0.71	0.79	0.82	**0.88**
	ψ	0.17	0.14	0.10	**0.08**	0.09
f19	α	0.74	0.79	0.85	0.87	**0.91**
	ψ	0.14	0.11	0.08	**0.07**	0.08
f20	α	0.61	0.67	0.73	0.76	**0.83**
	ψ	0.19	0.15	0.12	**0.09**	0.10
f21	α	0.68	0.72	0.79	0.81	**0.87**
	ψ	0.16	0.13	0.09	**0.08**	0.09
f22	α	0.70	0.76	0.82	0.85	**0.89**
	ψ	0.15	0.11	0.08	**0.07**	0.08
f23	α	0.62	0.68	0.74	0.78	**0.84**
	ψ	0.18	0.14	0.11	**0.09**	0.10

Table 5. Accuracy (α) and Precision (ψ) metrics for fixed-multimodal benchmark functions with $T_{max} = 1000$ and $T_{exp} = 30$

these functions. DE also tends to perform well with respect to **precision (ψ)**, although there are cases where other algorithms, such as **QPSO** and **DB-PSO**, outperform DE in terms of precision. Specifically, QPSO has lower ψ values than DE for functions f_{16}, f_{17}, and f_{20}, indicating higher precision in these cases.

Overall, **DE** is the most accurate algorithm for these fixed-multimodal functions, making it a strong choice when accuracy is the priority. **QPSO** and **DB-PSO**, however, excel in precision, offering more consistent results across multiple runs. These differences highlight the trade-off between accuracy and precision in choosing the appropriate optimization algorithm for specific tasks.

Each algorithm was run independently 30 times per function to account for stochastic variability, and performance was evaluated using the best fitness value obtained, convergence speed, and consistency across runs. The parameters for all algorithms were tuned to ensure fairness in comparison. Results indicate that the Gradient-Perturbation PSO and Dual-Binary PSO consistently outperformed standard PSO in terms of solution quality and robustness, particularly on complex multimodal functions. These findings highlight the effectiveness of incorporating gradient-based local search and binary-inspired mechanisms for enhancing global search capabilities.

Conclusion: All experiments were implemented in Python and executed on Google Colab, leveraging cloud-based computation with access to high-performance virtual machines. Each algorithm was independently run 30 times per benchmark function to ensure statistical reliability and account for stochastic variability. The algorithms compared in this study include: Standard PSO (serving as the baseline), Gradient-Perturbation PSO (a proposed enhancement), and Dual-Binary PSO (a novel variant).

Case study: benchmark functions

See Tables 6,7 and 8

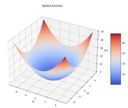
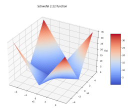
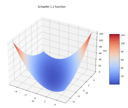
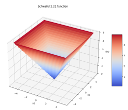
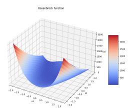
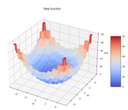
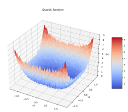
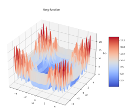
Name	Function	Range	$f_{\min}$	$n = 2$ representation
Sphere	$f_1(x) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^n$	0	
Schwefel's No. 2.22	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-10, 10]^n$	0	
Schwefel's No. 1.2	$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]^n$	0	
Schwefel's No. 2.21	$f_4(x) = \max\left(\lvert x_i \rvert, 1 \leq i \leq n \right)$	$[-100, 100]^n$	0	
Rosenbrock	$f_5(x) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-30, 30]^n$	0	
Step	$f_6(x) = \sum_{i=1}^{n} \left(\lfloor x_i + 0.5 \rfloor \right)^2$	$[-100, 100]^n$	0	
Quartic	$f_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}(0, 1)$	$[-1.28, 1.28]^n$	0	
X.S. Yang No. 7	$f_8(x) = \sum_{i=1}^{n} \epsilon_i \lvert x_i \rvert, \quad \epsilon_i \in [0, 1]$	$[-5, 5]^n$	0	

Table 6. Unimodal benchmark functions.

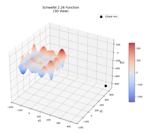
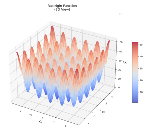
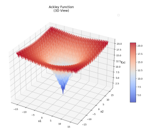
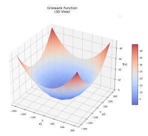
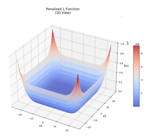
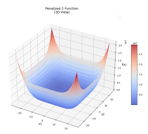
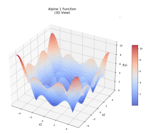
Name	**Function**	**Range**	$f_{\min}$	**2D Plot**
Schwefel No. 2.26	$f_9(x) = -\sum x_i \sin(\sqrt{\lvert x_i \rvert})$	$[-500, 500]^n$	≈ -12569.5	
Rastrigin	$f_{10}(x) = \sum [x_i^2 - 10\cos(2\pi x_i) + 10]$	$[-5.12, 5.12]^n$	0	
Ackley	$f_{11}(x) = -20e^{-0.2\sqrt{\frac{1}{n}\sum x_i^2}} - e^{\frac{1}{n}\sum \cos(2\pi x_i)} + 20 + e$	$[-32, 32]^n$	0	
Griewank	$f_{12}(x) = \sum \frac{x_i^2}{4000} - \prod \cos(x_i/\sqrt{i}) + 1$	$[-600, 600]^n$	0	
Penalized No. 1	$f_{13}(x) = \frac{\pi}{n}[10\sin^2(\pi y_1) + \sum (y_i - 1)^2(1 + 10\sin^2(\pi y_{i+1})) + (y_n - 1)^2] + \sum u(x_i, 10, 100, 4)$	$[-50, 50]^n$	0	
Penalized No. 2	$f_{14}(x) = 0.1[\sin^2(3\pi x_1) + \sum (x_i - 1)^2(1 + \sin^2(3\pi x_{i+1})) + (x_n - 1)^2(1 + \sin^2(2\pi x_n))] + \sum u(x_i, 5, 100, 4)$	$[-50, 50]^n$	0	
Alpine No. 1	$f_{15}(x) = \sum \lvert x_i \sin(x_i) + 0.1x_i \rvert$	$[-10, 10]^n$	0	

Table 7. Multimodal benchmark functions commonly used for optimization performance assessment. Each function's global minimum is provided alongside its domain.

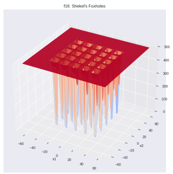
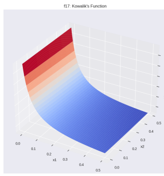
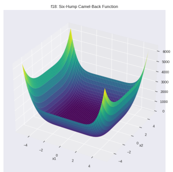
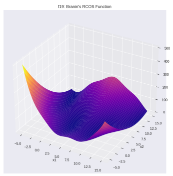
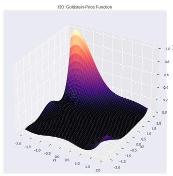
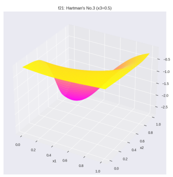
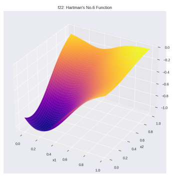
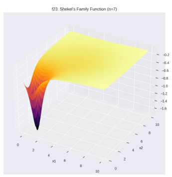
Name	Function	Range	$f_{\min}$	$n = 2$ rep.
Shekel's Foxholes	$f_{16}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j+(x_i - a_{ij})^6}\right]^{-1}$	$[-65, 65]^{10}$	1	
Kowalik	$f_{17}(x) = \sum_{i=1}^{11} \left(a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right)^2$	$[-5, 5]^4$	0.00031	
Six-Hump Camel-Back	$f_{18}(x) = (4x_1^2 - 2.1x_1^4 + \frac{x_1^6}{3})x_1^2 + x_1 x_2$ $-4x_2^2 + 4x_2^4$	$[-5, 5]^2$	-1.03163	
Branin's RCOS 1	$f_{19}(x) = \left(x_2 - \frac{5.1x_1^2}{4\pi^2} + \frac{5x_1}{\pi} - 6\right)^2 +$ $10(1 - \frac{1}{8\pi}\cos(x_1) + 10$	$[-5, 15]^2$	0.39800	
Goldstein-Price	$f_{20}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1$ $+3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]$ $\times[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2$ $+48x_2 - 36x_1x_2 + 27x_2^2)]$	$[-2, 2]^2$	3	
Hartman's No. 3	$f_{21}(x) = -\sum_{i=1}^{4} c_i \exp[-\sum_{j=1}^{3} a_{ij}(x_j -$ $p_{ij})^2]$	$[0, 1]^3$	-3.8628	
Hartman's No. 6	$f_{22}(x) = -\sum_{i=1}^{4} c_i \exp[-\sum_{j=1}^{6} a_{ij}(x_j -$ $p_{ij})^2]$	$[0, 1]^6$	-3.32	
Shekel No. 7	$f_{23}(x) = -\sum_{i=1}^{5}\left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	$[0, 10]^4$	-10.4028	

Table 8. Fixed-dimension multimodal benchmark functions.

References

[1] Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995. [Google Scholar]

[2] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh and S. Mirjalili, "Particle Swarm Optimization: A Comprehensive Survey," in IEEE Access, vol. 10, pp. 10031-10061, 2022, doi: 10.1109/ACCESS.2022.3142859.

[3] M. Shami, Tareq & El-Saleh, Ayman & Alswaitti, Mohammed & Al-Tashi, Qasem & Summakieh, Amen & Mirjalili, Seyedali. (2022). Particle Swarm Optimization: A Comprehensive Survey. IEEE Access. PP. 1-1. 10.1109/ACCESS.2022.3142859.

[4] Lei Wu, Xiaodong Huang, Junguo Cui, Chao Liu, Wensheng Xiao, Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot, Expert Systems with Applications, Volume 215, 2023, 119410, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2022.119410.

[5] Shao, J., Lu, Y., Sun, Y. et al. An improved multi-objective particle swarm optimization algorithm for the design of foundation pit of rail transit upper cover project. Sci Rep 15, 10403 (2025). https://doi.org/10.1038/s41598-025-87350-8

[6] Alvarez-Alvarado, M.S., Alban-Chacón, F.E., Lamilla-Rubio, E.A. et al. Three novel quantum-inspired swarm optimization algorithms using different bounded potential fields. Sci Rep 11, 11655 (2021). https://doi.org/10.1038/s41598-021-90847-7

[7] Zhu, D., Li, R., Zheng, Y. et al. Cumulative Major Advances in Particle Swarm Optimization from 2018 to the Present: Variants, Analysis and Applications. Arch Computat Methods Eng 32, 1571–1595 (2025). https://doi.org/10.1007/s11831-024-10185-5

[8] M. A. Kamel, X. Yu and Y. Zhang, "Real-Time Fault-Tolerant Formation Control of Multiple WMRs Based on Hybrid GA–PSO Algorithm," in IEEE Transactions on Automation Science and Engineering, vol. 18, no. 3, pp. 1263-1276, July 2021, doi: 10.1109/TASE.2020.3000507

[9] Kadkol, A.A. (2021). Mathematical Model of Particle Swarm Optimization: Numerical Optimization Problems. In: Mercangöz, B.A. (eds) Applying Particle Swarm Optimization. International Series in Operations Research Management Science, vol 306. Springer, Cham. https://doi.org/10.1007/978-3-030-70281-65

[10] M. Taherkhani and R. Safabakhsh, *A novel stability-based adaptive inertia weight for particle swarm optimization*, Applied Soft Computing, vol. 38, pp. 281–295, 2016.

[11] K. Chen, F. Zhou, and A. Liu, *Chaotic dynamic weight particle swarm optimization for numerical function optimization*, Knowledge-Based Systems, vol. 139, pp. 23–40, 2018

[12] James Stuart Groves, *A study of stochastic processes in Banach spaces* PhD thesis, pp.9-16

[13] Xionghua Huang, Tiaojun Zeng , and MinSong Li,*A Particle Swarm Optimization Algorithm with Gradient Perturbation and Binary Tree Depth First Search Strategy*, https://doi.org/10.1155/2022/6599899,2022

[14] Temlyakov V. *On the rate of convergence of Greedy Algorithms* Mathematics 2023, 11, 2559. https://doi.org/10.3390/math11112559

[15] J. Lui, Y. Mei, X. Li *An analysis of the Inertia weight parameter for binary Particle Swarm Optimization*, IEEE, 2016

[16] Song, Y.; Lui Y; Chen H?; Dend W. *A Multi-strategy adaptive particle swarm optimization* Electronics 2023.

[17] X. Chen and Y. Li *A modified particle swarm optimization structure resulting in high exploration ability with convergence guaranteed* IEE Trans. Syst. Man Cybern. B Cybern., vol. 37, no. 11, pp. 1591-1604, 2003

[18] Q. Yuan and G. Yin *Analyzing convergence and rates of convergence of particle swarm algortihms using stochastic approximation methods* IEE arXiv:1308.5369v2(2014)

[19] Temlyakov, V.N. Greedy Expansions in Banach spaces. adv compt. Math.2007, 26, 431-449

[20] Chun-Yao Lee, Truong-an Le *An enhanced Binary particle swarm optimizatytion for optimial feature sekection in bearing fault diagnosis of electrical machines* IEE (2021)

[21] M. Berasategui, S. Lassalle, *Weak greedy algorithms and equivalance between semi-greedy and alsmost greedy Markushevich bases* arXiv:2004.06849v4 (2021)

[22] A.A. Khartov, I.A. Limar, *Asymptotic analysis in multivariate worst case approximation with Gaussian kernels,* Journal of Complexity, Volume 82, 2024, 101838, ISSN 0885-064X, https://doi.org/10.1016/j.jco.2024.101838.