Metaheuristic Algorithms with Applications

www.maa.reapress.com

Metaheur. Algor. Appl. Vol. 1, No. 3 (2024) 316–341.

Paper Type: Original Article

# A Multi-Strategy Population-Based Optimizer for High-Dimensional Engineering Design Problems

**G. T. Shakila Devi***

Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences (SIMATS), Saveetha University, Chennai, Tamil Nadu, India; shakiladevigt.sse@saveetha.com.

**Citation:**

| | |
|---|---|
| *Received: 22 January 2024*<br>*Revised: 13 March 2024*<br>*Accepted: 08 June 2024* | Devi, G. T. Sh. (2024). A multi-strategy population-based optimizer for high-dimensional engineering design problems. *Metaheuristic algorithms with applications, 1*(3), 316-341. |

**Abstract**

Metaheuristic optimization algorithms are indispensable tools for solving complex engineering design problems, yet most existing methods suffer from premature convergence and an inadequate exploration–exploitation balance when applied to high-dimensional search spaces. The curse of dimensionality causes exponential growth of the feasible region, rendering algorithms that perform well at 30–50 dimensions increasingly ineffective at 100 dimensions and beyond. This paper proposes the Multi-Strategy Population-based Optimizer (MSPO), a novel metaheuristic that addresses these challenges through three synergistic search strategies coordinated by an adaptive selection controller. Strategy 1, Lévy-Enhanced Global Exploration (LEGE), employs Lévy flight-driven position updates with adaptive, dimensionality-scaled step sizes to enable long-range exploration of multimodal landscapes. Strategy 2, Opposition-Based Directional Exploitation (OBDE), combines opposition-based learning with a multi-reference directional exploitation vector that targets a convex combination of the global best, local neighborhood best, and elite centroid, thereby avoiding single-attractor stagnation. Strategy 3, Stochastic Dimensional Crossover (SDC), performs dimension-wise recombination with adaptive Bernoulli masks driven by per-dimension population variance, preserving diversity in unconverged dimensions while accelerating convergence in settled ones. A dynamic strategy selection controller activates strategies based on real-time population diversity, fitness improvement rate, and individual stagnation indicators. Critical dimensionality-scaling mechanisms automatically adjust strategy parameters for problems ranging from 10 to 1000 dimensions. MSPO is comprehensively evaluated on the CEC 2017 benchmark suite (30D, 50D, 100D), the CEC 2022 suite (10D, 20D), extended high-dimensional tests (500D, 1000D), and eight constrained real-world engineering design problems against 12 state-of-the-art algorithms. MSPO achieves the top Friedman rank across all benchmark suites, with the performance advantage widening significantly at higher dimensions, attaining 2–5 orders of magnitude better convergence accuracy than the best competitor at 500D and 1000D. On all eight engineering design problems, MSPO finds optimal or best-known feasible solutions with the lowest variance across independent runs.

**Keywords:** Metaheuristic optimization, Multi-strategy optimizer, High-dimensional optimization, Engineering design, Exploration–exploitation balance, Population-based algorithm, Lévy flight, Opposition-based learning.

## 1 | Introduction

Optimization lies at the heart of modern engineering design. From determining the minimum-weight cross-sections of structural members in civil and aerospace frameworks to calibrating process parameters in chemical plants, and from sizing electrical circuits for minimal power dissipation to shaping aerodynamic profiles for maximum lift-to-drag ratios, engineers routinely face the task of finding the best feasible combination of design variables subject to physical, economic, and regulatory constraints [1]. Historically,

Corresponding Author: shakiladevigt.sse@saveetha.com

https://doi.org/10.48313/maa.v1i3.103

gradient-based methods such as sequential quadratic programming and interior-point algorithms have served as the primary optimization workhorses. However, the increasing complexity, non-linearity, non-convexity, and multimodality of contemporary engineering objective functions have exposed fundamental limitations of classical approaches, including sensitivity to initial points, inability to escape local optima, and the requirement for continuous, differentiable objective and constraint functions [2].

Over the past three decades, metaheuristic optimization algorithms have emerged as versatile alternatives that impose minimal assumptions on the problem structure. These population-based stochastic search methods draw inspiration from diverse natural and physical phenomena. They can be broadly categorized into several families: Swarm Intelligence (SI) algorithms such as Particle Swarm Optimization (PSO) [3], Grey Wolf Optimizer (GWO) [4], Whale Optimization Algorithm (WOA) [5], Salp Swarm Algorithm (SSA) [6], Harris Hawks Optimization (HHO) [7], and Marine Predators Algorithm (MPA) [8]; Evolutionary Algorithms (EAs) including Genetic Algorithms (GA) [9], Differential Evolution (DE) [10], and the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [11]; and physics-based and mathematically inspired algorithms such as the Aquila Optimizer (AO) [12], Arithmetic Optimization Algorithm (AOA) [13], INFO [14], and Rime Optimization Algorithm (RIME) [15]. Comprehensive taxonomies and surveys of these methods are provided by Dokeroglu et al. [16], Yang [2], and Del Ser et al. [17].

Despite their proven success on a wide range of optimization problems, most metaheuristic algorithms are designed, tuned, and benchmarked on problems with at most 30 to 50 decision variables. In practice, however, many engineering design problems involve dozens to hundreds of design variables; finite element models may require optimizing the thickness of hundreds of shell elements, neural network hyperparameter tuning involves scores of continuous and categorical variables, and supply chain scheduling can encompass thousands of binary decisions. When the dimensionality D grows beyond 100, the volume of the search space increases exponentially, a phenomenon known as the curse of dimensionality [18]. This exponential scaling has severe consequences for population-based optimizers: the population becomes an increasingly sparse sample of the search space, pairwise distances between individuals grow, gradient information estimated from fitness differences becomes noisy, and the probability of a random perturbation yielding an improving move diminishes rapidly [19].

Examination of existing algorithms reveals systematic performance degradation at high dimensions. PSO suffers from velocity explosion and swarm collapse as the momentum term accumulates errors across many dimensions [20]. GWO's hierarchical leadership structure loses effectiveness because the alpha, beta, and delta wolves' position vectors become poor approximations of the true optimum in high-dimensional spaces [21]. WOA's spiral updating equation generates excessively large perturbations when D is large, causing slow convergence [22]. DE's Crossover Rate (CR) becomes a critical, dimension-sensitive parameter: too low a CR fails to propagate improvements across dimensions, while too high a CR disrupts partially converged solutions [23]. CMA-ES, though theoretically elegant, requires $O(D^2)$ memory for the covariance matrix and $O(D^3)$ computation per matrix decomposition, rendering it impractical beyond approximately 500 dimensions without decomposition heuristics [11]. Recent algorithms such as RIME, INFO, and MPA show promising performance on CEC benchmarks up to 100D, but have not been systematically evaluated at 500D or 1000D, and their internal mechanisms lack explicit dimensionality-scaling provisions.

The exploration–exploitation dilemma, central to all metaheuristic design, intensifies with dimensionality. In low dimensions, a moderate balance between global search (exploration) and local refinement (exploitation) suffices because the search space is manageable. In high dimensions, the algorithm must explore far more aggressively to maintain population diversity, yet it must simultaneously exploit promising regions with greater precision because the number of potential improvement directions grows linearly with D. A static, predetermined balance, whether hardcoded into the algorithm's equations or controlled by a monotonically varying parameter, inevitably fails as dimensionality increases [24]. Multi-strategy approaches that combine different search operators offer a promising direction, as exemplified by Multi-Strategy Biogeography-Based Optimization (MSBBO) [25], Linear Success-History based Adaptive DE with Semi-Parameter Adaptation

(L-SHADE-SPA) [26], and Multi-strategy Improved Honey Badger Algorithm (MIHBA) [27]. However, most existing multi-strategy methods employ fixed operator allocation schedules or simplistic random selection, lacking a principled mechanism that adapts operator deployment based on real-time population state indicators and that explicitly accounts for problem dimensionality.

Motivated by these gaps, this paper proposes the Multi-Strategy Population-based Optimizer (MSPO), a novel metaheuristic algorithm specifically designed for high-dimensional engineering design optimization. The principal contributions of this work are as follows:

I. MSPO algorithm: a novel three-strategy metaheuristic integrating Lévy-Enhanced Global Exploration (LEGE), Opposition-Based Directional Exploitation (OBDE), and Stochastic Dimensional Crossover (SDC), each addressing a distinct aspect of the search process.

II. Adaptive strategy selection controller: a dynamic mechanism that assigns strategies to individuals based on population diversity index, fitness improvement rate, and individual stagnation counters, enabling real-time, situation-aware exploration–exploitation balance without manual tuning.

III. Dimensionality-scaling parameter adaptation: strategy parameters including Lévy step size, crossover probability, and stagnation thresholds automatically scale with problem dimensionality D, enabling effective optimization from 10D to 1000D without user intervention.

IV. Comprehensive benchmarking: evaluation on the CEC 2017 suite (30D, 50D, 100D), CEC 2022 suite (10D, 20D), and extended high-dimensional tests (500D, 1000D) against 12 state-of-the-art algorithms, providing the most extensive dimensionality coverage reported for a single metaheuristic.

V. Engineering validation: application to eight constrained real-world engineering design problems, demonstrating practical applicability and superiority over all compared methods.

The remainder of this paper is organized as follows. Section 2 reviews related work on metaheuristic optimization with emphasis on high-dimensional challenges and multi-strategy approaches. Section 3 details the MSPO methodology, including mathematical formulations, pseudocode, and complexity analysis. Section 4 describes the experimental setup, benchmark suites, engineering problems, compared algorithms, and statistical testing procedures. Section 5 presents and analyzes the experimental results. Section 6 discusses the findings, implications, and limitations. Section 7 concludes the paper and outlines future research directions.

# 2 | Related Work

## 2.1 | Swarm Intelligence Algorithms

SI algorithms model the collective behavior of decentralized, self-organized agents. PSO [3] remains the most widely adopted SI algorithm, with numerous variants addressing its limitations: Comprehensive Learning Particle Swarm Optimization (CLPSO) [28] uses a multi-exemplar learning strategy that improves multimodal performance; Genetic Learning Particle Swarm Optimization (GLPSO) [29] hybridizes PSO with genetic operators; and Social Learning-Particle Swarm Optimization (SL-PSO) [30] introduces a social learning mechanism particularly effective for Large-Scale Optimization (LSO). GWO [4] simulates the hunting hierarchy of grey wolves, offering simplicity and competitive performance on low-dimensional problems but suffering from limited diversity maintenance in high dimensions due to its reliance on three leadership positions. WOA [5] models the bubble-net hunting strategy of humpback whales, featuring spiral-updating and encircling prey mechanisms, but the spiral equation generates excessively large perturbations at high D. SSA [6] emulates salp chain behavior in oceans, showing strong exploitation but limited exploration capability. HHO [7] mimics Harris hawks' cooperative hunting, using soft and hard besiege strategies with good balance in moderate dimensions but diminishing returns beyond 100D. MPA [8] is inspired by marine predators' foraging strategies using Lévy and Brownian motion, showing competitive CEC performance. The AO [12] simulates the hunting behavior of Aquila eagles, incorporating four search strategies, and the Sine Cosine Algorithm (SCA) [31] uses sine and cosine functions for position updates. While these algorithms demonstrate excellent performance on standard 30D benchmarks, systematic studies reveal significant performance

degradation beyond 100D, primarily due to insufficient population diversity, fixed exploration–exploitation transition schedules, and the absence of dimensionality-aware parameter scaling.

## 2.2 | Evolutionary Algorithms

EAs are among the oldest and most well-studied metaheuristics. GA [9] uses selection, crossover, and mutation operators inspired by biological evolution. Real-coded GAs with Simulated Binary Crossover (SBX) and polynomial mutation [32] are widely used for continuous optimization, though crossover disruption becomes increasingly problematic in high dimensions where beneficial building blocks span many loci. DE [10] operates through mutation, crossover, and selection on real-valued vectors and has become the dominant EA for continuous optimization, spawning numerous successful variants: adaptive DE with Optional External Archive (JADE) [33] introduces adaptive parameter control; SHADE [34] extends JADE with a historical memory of successful parameters; L-SHADE [35] adds Linear Population Size Reduction (LPSR), achieving top rankings in CEC competitions; and L-SHADE-RSP [36] further refines the restart and parameter adaptation mechanisms. DE variants are currently the strongest competitors in CEC competitions, particularly at 30–50D. CMA-ES [11] adapts the full covariance matrix of a multivariate normal search distribution, providing theoretically optimal step-size adaptation on convex functions. However, its $O(D^2)$ space and $O(D^3)$ per-generation time complexity render it impractical for $D > 500$ without decomposition approaches such as sep-CMA-ES or VD-CMA [37], which sacrifice the ability to capture variable interactions.

## 2.3 | Recent Novel Metaheuristics

The past five years have witnessed a proliferation of novel metaheuristic algorithms with increasingly creative metaphors and sophisticated mechanisms. RIME [15] simulates rime-ice formation with hard-rime and soft-rime search phases, demonstrating competitive CEC 2017 performance. INFO [14] uses a weighted mean of vectors for position updating, achieving strong results on diverse function types. The Growth Optimizer (GO) [38] models plant growth patterns with learning and reflection phases. The Snow Ablation Optimizer (SAO) [39] simulates snow melting dynamics. The Parrot Optimizer (PO) [40] models foraging and communication behaviors of parrots. While these algorithms introduce novel search operators and often demonstrate competitive or superior performance on CEC benchmarks at standard dimensions, they share a common limitation: their evaluation is typically restricted to $D \leq 100$, and their internal mechanisms lack explicit provisions for scaling to very high dimensions. Furthermore, the No Free Lunch theorem [41] guarantees that no single algorithm outperforms all others across all possible problems, motivating the development of multi-strategy approaches that can adapt their search behavior to problem characteristics.

## 2.4 | Multi-Strategy and Adaptive Approaches

Multi-strategy optimization combines multiple search operators within a single algorithmic framework, leveraging the complementary strengths of different strategies. MSBBO [25] enhances biogeography-based optimization with multiple migration and mutation strategies for high-dimensional problems. L-SHADE-SPA [26] augments L-SHADE with semi-parameter adaptation for improved CEC performance. MIHBA [27] integrates multiple improvement strategies into the honey badger algorithm. Adaptive Operator Selection (AOS) [42]: methods dynamically choose among a portfolio of operators based on their recent performance, using credit assignment and selection rules. Ensemble methods such as EPSDE [43] maintain pools of parameter configurations and operator combinations, adapting the ensemble composition based on success rates. Self-adaptive DE variants [44], [45] evolve control parameters alongside solution vectors. While these approaches represent significant advances, most employ either fixed allocation schedules, simple success-rate-based selection, or credit assignment mechanisms that respond to individual operator success without considering the global population state (diversity, stagnation, improvement momentum). Furthermore, none incorporate explicit dimensionality-scaling mechanisms for their strategy parameters.

## 2.5 | High-Dimensional and Large-Scale Optimization

LSO has received growing attention, formalized through the CEC 2013 Large-Scale Global Optimization (LSGO) competition [46] with problems up to 1000D. The dominant paradigm for LSO is Cooperative Coevolution (CC), introduced by Potter and De Jong [47] and advanced by Yang et al. [48], which decomposes the high-dimensional problem into lower-dimensional subproblems optimized in alternation. Decomposition methods include random grouping [48], differential grouping [19], and recursive differential grouping [49]. While CC methods can handle 1000D+ problems, they require accurate identification of variable interactions, a task that itself becomes computationally expensive, and their performance degrades when strong interactions exist between many variables. Direct-search approaches that optimize all dimensions simultaneously avoid the decomposition requirement but face the full brunt of the curse of dimensionality. Notable direct-search methods for LSO include SL-PSO [30], multiple-strategy self-adaptive DE (JDE21; [50]), and competitive swarm optimizer (CSO; [51]). The present work proposes MSPO as a direct-search method with built-in dimensionality-scaling mechanisms that enable effective optimization up to 1000D without problem decomposition.

## 2.6 | Engineering Design Optimization

Constrained engineering design optimization problems serve as important benchmarks for evaluating metaheuristic algorithms on practical applications. Classic test problems include welded beam design [52], pressure vessel design [53], tension/compression spring design [54], speed reducer design [55], three-bar truss design [56], cantilever beam design [57], gear train design [58], and rolling element bearing design [59]. These problems feature nonlinear objective functions, nonlinear inequality and equality constraints, mixed continuous–discrete variables, and search spaces ranging from 2D to 10D. Constraint handling techniques include static and dynamic penalty methods [60], Deb's feasibility rules [61], ε-constrained methods [62], and stochastic ranking [63]. Recent meta-analyses compile best-known solutions across dozens of algorithms [64], providing reference points for comparison. While individual engineering problems are relatively low-dimensional, they test an algorithm's ability to navigate constraint boundaries and find globally optimal feasible solutions capabilities that are prerequisite for tackling larger real-world engineering optimization tasks.

# 3 | Proposed Multi-Strategy Population-based Optimizer Algorithm

## 3.1 | Problem Definition

Without loss of generality, MSPO is formulated for the general constrained minimization problem:

$$\text{Minimize } f(x), \quad x = [x_1, x_2, \ldots, x_D] \in \mathbb{R}^D. \tag{1}$$

subject to:

$$g_j(x) \le 0, \quad j = 1, 2, \ldots, J \quad \text{(inequality constraints)}, \tag{2}$$

$$|h_k(x)| \le \delta, \quad k = 1, 2, \ldots, K \quad \text{(equality constraints, relaxed)}, \tag{3}$$

$$x_i^L \le x_i \le x_i^U, \quad i = 1, 2, \ldots, D \quad \text{(bound constraints)}, \tag{4}$$

where $f : \mathbb{R}^D \to \mathbb{R}$ is the objective function, J and K are the numbers of inequality and equality constraints, respectively, $\delta = 10^{-4}$ is the equality constraint tolerance, and $x_i^L$ and $x_i^U$ are the lower and upper bounds of the i-th decision variable.

## 3.2 | Population Initialization via Chaotic Tent Map

High-dimensional search spaces amplify the impact of initial population distribution on algorithmic performance. Pseudo-random initialization, while statistically uniform in expectation, exhibits clustering and

gaps that worsen with increasing dimensionality [65]. MSPO employs the Tent map chaotic sequence for initialization, which provides more uniform coverage of the search space through deterministic chaos:

$$z_{n+1} = z_n / 0.7, \quad \text{if } z_n < 0.7, \tag{5}$$
$$z_{n+1} = (1 - z_n) / 0.3, \quad \text{if } z_n \geq 0.7. \tag{6}$$

For each dimension d = 1, 2, ..., D, an independent chaotic sequence is generated with a distinct initial value $z_0^{(d)}$ drawn uniformly from (0, 1), avoiding the fixed points at 0 and 1. The i-th individual's d-th coordinate is then mapped to the decision space:

$$x_{i,d} = x_d^L + z_i^{(d)} \cdot (x_d^U - x_d^L), \quad i = 1, 2, ..., N, \tag{7}$$

where N is the population size. This chaotic initialization has been shown to improve space-filling properties and enhance early-stage exploration, particularly in high-dimensional spaces where the discrepancy of pseudo-random sequences becomes non-negligible [66].

## 3.3 | Strategy 1: Lévy-Enhanced Global Exploration

The first strategy is designed for global exploration of the search space, using Lévy flights to generate position updates with heavy-tailed step-size distributions. The heavy-tailed property of the Lévy distribution ensures that most steps are short (enabling local refinement of explored regions) while occasional long jumps enable the algorithm to escape local optima and reach distant, unexplored regions of the search space [67].

The position update rule for individual i under LEGE is:

$$x_i^{new} = x_i + \alpha(t) \cdot \text{Lévy}(\beta) \otimes (x_i - x_{best}), \tag{8}$$

where $\otimes$ denotes element-wise multiplication, $x_{best}$ is the global best position, and Lévy(β) is a vector of D independent Lévy-distributed random numbers with stability index β = 1.5. The Lévy random numbers are generated using Mantegna's algorithm [68]:

$$s = u / |v|^{1/\beta}, \quad u \sim N(0, \sigma_u^2), \quad v \sim N(0, 1), \tag{9}$$
$$\sigma_u = [\, \Gamma(1 + \beta) \cdot \sin(\pi\beta/2) / ( \Gamma((1 + \beta)/2) \cdot \beta \cdot 2^{(\beta-1)/2} ) \,]^{1/\beta}. \tag{10}$$

The adaptive step size α(t) incorporates four critical factors: an initial scale, dimensionality scaling, temporal decay, and diversity responsiveness:

$$\alpha(t) = \alpha_0 \cdot D^{-0.5} \cdot \exp(-c_1 \cdot t / T) \cdot (1 + c_2 \cdot D(t)), \tag{11}$$

where $\alpha_0 = 1.0$ is the initial step scale, $D^{-0.5}$ scales the step size inversely with the square root of dimensionality (preventing excessively large steps in high-D spaces), $c_1 = 2.0$ is the temporal decay rate ensuring gradual transition from exploration to exploitation, $c_2 = 0.5$ is the diversity boost coefficient that increases step size when population diversity D(t) is low (counteracting premature convergence), t is the current generation, and T is the maximum number of generations. The $D^{-0.5}$ scaling factor is the critical innovation for high-dimensional stability. In a D-dimensional space, the expected Euclidean norm of a random perturbation vector scales as √D, so dividing by √D normalizes the expected step length to be dimension-independent.

## 3.4 | Strategy 2: Opposition-Based Directional Exploitation

The second strategy focuses on exploiting promising regions while maintaining the ability to escape local optima through opposition-based learning [69]. Unlike standard global-best-based exploitation that pulls all individuals toward a single attractor (risking swarm collapse), OBDE uses a multi-reference target point that balances information from multiple sources.

**Step 1.** Multi-reference target computation. For individual i, the target point is a convex combination of three reference positions:

$$x_{target} = w_1 \cdot x_{best} + w_2 \cdot x_{lbest,i} + w_3 \cdot x_{centroid,K}, \quad (12)$$

where $x_{best}$ is the global best solution, $x_{lbest,i}$ is the best solution in the ring neighborhood of individual i (neighborhood size = 5), $x_{centroid,K}$ is the centroid of the top K = max(5, ⌊N/10⌋) elite solutions, and $w_1 = 0.5$, $w_2 = 0.3$, $w_3 = 0.2$ are the default weights satisfying $w_1 + w_2 + w_3 = 1$. This multi-reference scheme provides a more robust exploitation direction than single-attractor methods: $x_{best}$ provides global information, $x_{lbest,i}$ provides local topological information, and $x_{centroid,K}$ provides a smoothed estimate of the elite region's center.

**Step 2.** Directional exploitation. The exploitation candidate is generated as:

$$x_i^{exploit} = x_i + r \cdot (x_{target} - x_i) \cdot \exp(-t / T), \quad (13)$$

where $r \sim U(0, 1.5)$ is a uniform random scalar. The factor $\exp(-t/T)$ ensures that the step size decreases over generations, refining the search as the algorithm converges. Values of $r > 1$ allow the individual to overshoot the target, exploring the region beyond the convex combination.

**Step 3.** Opposition-based reflection. The opposition candidate is generated using generalized opposition-based learning:

$$x_i^{opp} = x^L + x^U - x_i^{exploit}. \quad (14)$$

The final updated position is selected as the better candidate:

$$x_i^{new} = \text{argmin}\{ f(x_i^{exploit}), f(x_i^{opp}) \}. \quad (15)$$

This dual candidate generation gives each individual two chances to improve per application of Strategy 2: one through directed exploitation toward the multi-reference target, and one through opposition-based reflection. The opposition component is particularly valuable in high dimensions, where reflecting a solution across the search space center can discover distant improving regions that would be unreachable through local perturbation alone [70].

## 3.5 | Strategy 3: Stochastic Dimensional Crossover

The third strategy addresses a fundamental challenge in high-dimensional optimization: different dimensions converge at different rates. In a 1000-dimensional problem, some variables may reach near-optimal values early while others remain widely distributed. A uniform crossover operator that treats all dimensions identically either disrupts converged dimensions (wasting computation) or fails to explore unconverged ones (missing improvements). SDC addresses this with dimension-aware crossover using adaptive Bernoulli masks.

**Step 1.** Donor selection. Two individuals $x_a$ and $x_b$ are selected uniformly at random from the population, with $a \neq b \neq i$.

**Step 2.** Per-dimension variance computation. The population variance for each dimension d is computed as:

$$\sigma_d^2 = \text{Var}(\{x_{1,d}, x_{2,d}, \ldots, x_{N,d}\}), \quad d = 1, 2, \ldots, D. \quad (16)$$

**Step 3.** Adaptive Bernoulli mask generation. A binary mask $M = [m_1, m_2, \ldots, m_D]$ is generated where each element follows an independent Bernoulli distribution:

$$m_d \sim \text{Bernoulli}(p_d), \quad p_d = 1 - \sigma_d^2 / \max_d(\sigma_d^2). \quad (17)$$

This formulation ensures that converged dimensions (low $\sigma_d^2$) have a high crossover probability, $p_d \approx 1$ (encouraging exchange of potentially superior values), while diverse dimensions (high $\sigma d2$) have a low crossover probability, $p_d \approx 0$ (preserving the exploration in progress). The crossover probability is further scaled with dimensionality by applying a minimum threshold of $p_{min} = 1 / (1 + \ln(D))$ to ensure at least some dimensions are crossed even in high-D spaces.

**Step 4.** Offspring construction. The offspring is assembled dimension by dimension:

$$x_i^{new}[d] = x_a[d] + r \cdot (x_b[d] - x_a[d]), \quad \text{if } m_d = 1 \quad \text{(crossed)}, \tag{18}$$

$$x_i^{new}[d] = x_i[d], \quad \text{if } m_d = 0 \quad \text{(preserved)}, \tag{19}$$

where $r \sim U(-0.5, 1.5)$ allows both interpolation and extrapolation between donor values. This dimension-wise awareness is especially powerful in high-dimensional spaces where treating all dimensions uniformly fails to respect the heterogeneous convergence state of the population across dimensions.

## 3.6 | Dynamic Strategy Selection Controller

The dynamic strategy selection controller is the central coordinating mechanism of MSPO. It monitors the population state through three quantitative indicators and assigns strategies to individuals based on predefined rules that map population states to strategy distributions.

Population diversity index: the normalized diversity is computed as the average squared deviation of individuals from the population centroid, normalized by the search range:

$$D(t) = (1 / (N \cdot D)) \cdot \Sigma_{i=1}^{N} \Sigma_{d=1}^{D} (x_{i,d} - \bar{x}_d)^2 / (x_d^U - x_d^L)^2, \tag{20}$$

where $\bar{x}_d$ is the mean of the d-th dimension across all individuals.

Fitness improvement rate: the relative improvement in the best fitness over a sliding window of W = 20 generations:

$$R(t) = (f_{best}(t - W) - f_{best}(t)) / (|f_{best}(t - W)| + \varepsilon), \tag{21}$$

where $\varepsilon = 10^{-30}$ prevents division by zero.

Individual stagnation counter: for each individual i, $S_i$ counts the number of consecutive generations without fitness improvement.

Threshold calibration: rather than using fixed thresholds, MSPO auto-calibrates during the first 10% of generations by observing the initial diversity trajectory:

$$D_{low} = 0.1 \cdot D(0), \quad D_{high} = 0.5 \cdot D(0), \tag{22}$$

$$R_{low} = 0.001, \quad R_{high} = 0.01, \tag{23}$$

$$S_{threshold} = \max(20, \lfloor 0.1 \cdot T / D^{0.25} \rfloor). \tag{24}$$

The stagnation threshold $S_{threshold}$ scales with the ratio $T/D^{0.25}$, allowing more patience in higher dimensions where progress is inherently slower. Strategy assignment rules: At each generation, the population state is classified into one of three states, and strategies are assigned accordingly:

**State 1.** Low diversity, stagnating ($D(t) < D_{low}$ AND $R(t) < R_{low}$): the population has converged prematurely and is no longer improving. Assign LEGE to 60% of the population and SDC to 40%, triggering a diversification restart that re-explores the search space while maintaining some dimensional recombination.

**State 2.** High diversity, good progress ($D(t) > D_{high}$ AND $R(t) > R_{high}$): the population is well-spread and making good progress, the ideal conditions for intensified exploitation. Assign OBDE to 70% and LEGE to 30%, focusing search on the promising multi-reference target direction while maintaining minimal exploration.

**State 3.** Balanced (otherwise): neither extreme condition is met. Assign all three strategies equally (LEGE 33%, OBDE 33%, SDC 34%), maintaining a balanced search.

Individual override: any individual with $S_i > S_{threshold}$ must use LEGE regardless of the population state, providing a personal escape mechanism from stagnation.

**Algorithm 1. MSPO main loop.**

**Input:** Objective function f, dimension D, bounds $[x^L, x^U]$, population size N, max generations T **Output:** Global best solution $x_{best}$ and fitness $f_{best}$ 1: **Initialize** population $P = \{x_1, ..., x_N\}$ using Tent map chaotic sequences (Eqs. 5–7) 2: **Evaluate** $f(x_i)$ for all i = 1, ..., N 3: **Identify** $x_{best} = \text{argmin}_i f(x_i)$ 4: **Initialize** stagnation counters $S_i = 0$ for all i 5: **Initialize** sliding window buffer for fitness history 6: **Set** elite archive E = top 5% of P 7: **for** t = 1 to T **do** 8: **Compute** population diversity D(t) via Eq. 20 9: **Compute** fitness improvement rate R(t) via Eq. 21 10: **if** $t \leq 0.1 \cdot T$ **then** calibrate thresholds $D_{low}$, $D_{high}$ (Eq. 22) 11: **Classify** population state (A, B, or C) via strategy selection rules 12: **Assign** strategy to each individual via Algorithm 2 13: **for** i = 1 to N **do** 14: **if** assigned strategy = LEGE **then** 15: Generate $x_i^{new}$ via Eqs. 8–11 16: **else if** assigned strategy = OBDE **then** 17: Generate $x_i^{new}$ via Eqs. 12–15 18: **else if** assigned strategy = SDC **then** 19: Generate $x_i^{new}$ via Eqs. 16–19 20: **end if** 21: **Apply** boundary handling via reflection: 22: if $x_{i,d}^{new} < x_d^L$: $x_{i,d}^{new} = 2 \cdot x_d^L - x_{i,d}^{new}$ 23: if $x_{i,d}^{new} > x_d^U$: $x_{i,d}^{new} = 2 \cdot x_d^U - x_{i,d}^{new}$ 24: **Evaluate** $f(x_i^{new})$ 25: **if** $f(x_i^{new}) < f(x_i)$ **then** 26: $x_i \leftarrow x_i^{new}$; $S_i = 0$ 27: **else** 28: $S_i = S_i + 1$ 29: **end if** 30: **end for** 31: **Update** $x_{best}$ if improved 32: **Elitism:** Replace worst 5% of P with elite archive E 33: **Update** E = top 5% of current P 34: **end for** 35: **Return** $x_{best}$, $f_{best}$

**Algorithm 2. Dynamic strategy selection controller.**

**Input:** Diversity D(t), improvement rate R(t), stagnation counters $\{S_i\}$, thresholds $D_{low}$, $D_{high}$, $R_{low}$, $R_{high}$, $S_{threshold}$, population size N

**Output:** Strategy assignment vector $A = [a_1, ..., a_N]$

1: **Determine** population state:
2: **if** $D(t) < D_{low}$ AND $R(t) < R_{low}$ **then**
3: State ← A (Low diversity, Stagnating)
4: Assign LEGE to random 60% of {1,...,N}; SDC to remaining 40%
5:**else if** $D(t) > D_{high}$ AND $R(t) > R_{high}$ **then**
6: State ← B (High diversity, Good progress)
7: Assign OBDE to random 70% of {1,...,N}; LEGE to remaining 30%
8: **else**
9: State ← C (Balanced)
10: Assign LEGE to random 33%; OBDE to random 33%; SDC to remaining 34%
11: **end if**
12: **Individual override:**
13: **for** i = 1 to N **do**
14: **if** $S_i > S_{threshold}$ **then**
15: $a_i \leftarrow$ LEGE // Force exploration for stagnating individuals
16: $S_i \leftarrow 0$ // Reset stagnation counter
17: **end if**
18: **end for**
19: **Return** A

## 3.7 | Constraint Handling for Engineering Design Problems

For constrained optimization, MSPO employs Deb's feasibility rules [61] augmented with the ε-constrained method [62] for smooth constraint relaxation. The overall constraint violation for a solution x is:

$$CV(x) = \Sigma_{j=1}^{J} \max(0, g_j(x)) + \Sigma_{k=1}^{K} \max(0, |h_k(x)| - \delta). \quad (25)$$

The comparison rules between two solutions $x_a$ and $x_b$ are:

I. If $CV(x_a) = 0$ and $CV(x_b) = 0$ (both feasible): prefer the one with lower f value.

II. If $CV(x_a) = 0$ and $CV(x_b) > 0$ (one feasible, one infeasible): prefer the feasible solution.

III. If $CV(x_a) > 0$ and $CV(x_b) > 0$ (both infeasible): prefer the one with smaller CV.

The ε-level controls the tolerance for infeasibility and decays from an initial value $\varepsilon_0$ (set to the median initial population constraint violation) to 0 over the first 50% of generations:

$$\varepsilon(t) = \varepsilon_0 \cdot \max(0, 1 - 2t / T), \quad (26)$$

when $\varepsilon(t) > 0$, solutions with CV $\leq \varepsilon(t)$ are treated as feasible, allowing the population to explore infeasible regions near constraint boundaries during early generations. As $\varepsilon(t)$ decays to zero, the search transitions to strictly feasible optimization, ensuring that the final solution satisfies all constraints.

## 3.8 | Computational Complexity Analysis

The per-generation computational cost of MSPO is analyzed as follows:

I. Strategy 1 (LEGE): position update for each individual requires O(D) operations for vector arithmetic and Lévy number generation. For N individuals: O(N · D).

II. Strategy 2 (OBDE): multi-reference target computation requires O(K · D) for centroid calculation and O(D) for the weighted sum. Per-individual update: O(D). Opposition candidate evaluation adds one extra function evaluation. Total: O(N · D) plus N additional function evaluations.

III. Strategy 3 (SDC): per-dimension variance computation requires O(N · D). Bernoulli mask generation and offspring construction: O(D) per individual. Total: O(N · D).

IV. Diversity computation: O(N · D) for Eq. 20.

V. Elitist sorting: O(N log N) for identifying the top 5%.

VI. Strategy selection: O(N) for state classification and assignment.

The total per-generation complexity is O(N · D + N log N), dominated by O(N · D) for practical problem sizes. The total per-run complexity is O(T · N · D). The space complexity is O(N · D) for storing the population. *Table 1* compares the complexity of MSPO with key competitors.

**Table 1. Computational complexity comparison of MSPO with selected algorithms.**

| Algorithm | Time per Generation | Time per Run | Space Complexity |
|---|---|---|---|
| MSPO (proposed) | O(N·D) | O(T·N·D) | O(N·D) |
| PSO | O(N·D) | O(T·N·D) | O(N·D) |
| DE | O(N·D) | O(T·N·D) | O(N·D) |
| GWO | O(N·D) | O(T·N·D) | O(N·D) |
| L-SHADE | O(N·D) | O(T·N·D) | O(N·D+H) |
| CMA-ES | O(N·D2) | O(T·N·D2) | O(D2) |

MSPO shares the same asymptotic time complexity as PSO and DE, linear in both population size and dimensionality, while avoiding the quadratic space and time scaling of CMA-ES. The constant-factor overhead from diversity computation and strategy selection is modest (~30% in wall-clock time), as demonstrated in the experimental results.

# 4 | Experimental Setup

## 4.1 | Benchmark Test Suites

Three benchmark test suites are used to evaluate MSPO across a comprehensive range of dimensions and function characteristics:

I. CEC 2017 [71]: this suite comprises 29 benchmark functions (F1–F30, with F2 excluded due to instability) covering four categories: unimodal (F1, F3), simple multimodal (F4–F10), hybrid (F11–F20), and composition (F21–F30). The functions feature shifted optima, rotated coordinate systems, and complex interactions among variables. Testing is conducted at D = 30, 50, and 100 with a maximum number of function evaluations (MaxFEs) = 10,000 × D.

II. CEC 2022 [72]: the updated suite contains 12 benchmark functions covering unimodal, basic multimodal, hybrid, and composition categories. Testing is conducted at D = 10 and 20 with MaxFEs = 200,000.

III. Extended high-dimensional tests: ten representative functions from CEC 2017 (F1, F3, F4, F6, F9, F11, F15, F21, F25, F29) spanning all four categories are extended to D = 500 and D = 1000 with MaxFEs = 10,000 × D. These extreme-dimension tests assess scalability beyond the standard CEC evaluation regime.

## 4.2 | Engineering Design Problems

Eight well-known constrained engineering design problems are employed to evaluate MSPO's practical applicability:

**Table 2. Summary of engineering design benchmark problems.**

| ID | Problem | Variables | Constraints | Objective | Best Known |
|---|---|---|---|---|---|
| ED1 | Welded beam design | 4 | 7 | Min. fabrication cost | 1.72485 |
| ED2 | Pressure vessel design | 4 | 4 | Min. total cost | 5885.33 |
| ED3 | Tension/compression spring | 3 | 4 | Min. weight | 0.012665 |
| ED4 | Speed reducer design | 7 | 11 | Min. weight | 2994.471 |
| ED5 | Three-bar truss design | 2 | 3 | Min. weight | 263.8958 |
| ED6 | Cantilever beam design | 5 | 1 | Min. weight | 1.33996 |
| ED7 | Gear train design | 4 (integer) | 0 | Min. gear ratio error | $2.7009\times10^{-12}$ |
| ED8 | Rolling element bearing | 10 | 9 | Max. dynamic load | 83,269.78 |

## 4.3 | Compared Algorithms

MSPO is compared against 12 state-of-the-art algorithms spanning three categories:

I. Classic metaheuristics: PSO [3], DE/rand/1/bin [10], real-coded GA with SBX and polynomial mutation [32].

II. SI: GWO [4], WOA [5], SSA [6], HHO [7].

III. Recent algorithms: MPA [8], AOA [12], INFO [14], RIME [15].

IV. Competition-winning variant: L-SHADE [35].

## 4.4 | Parameter Settings

MSPO parameters are set as follows: population size N = 50 for D ≤ 100 and N = 100 for D > 100; maximum generations T = MaxFEs / N; initial step scale $\alpha_0 = 1.0$; Lévy index $\beta = 1.5$; temporal decay rate $c_1 = 2.0$; diversity boost coefficient $c_2 = 0.5$; sliding window size W = 20; elite count K = max(5, ⌊N/10⌋); ring neighborhood size = 5; elitism rate = 5%; ε-constraint initial value = median initial CV; ε decay over 50% of T. All diversity and improvement thresholds are auto-calibrated as described in Section 3.6. All compared algorithms use their default parameters as specified in the original publications, with population size N = 50 (for D ≤ 100) or N = 100 (for D > 100). For engineering design problems, MaxFEs = 30,000 for all algorithms. Each experiment is repeated for 30 independent runs.

## 4.5 | Statistical Tests

Pairwise comparisons employ the Wilcoxon rank-sum test at significance level p = 0.05 with Holm–Bonferroni correction for multiple comparisons [73]. Results are summarized using the symbols "+", "−", and "≈", denoting that MSPO is statistically significantly better, significantly worse, or statistically equivalent to the compared algorithm, respectively. The Friedman test is used for overall ranking across all benchmark functions within each test suite, providing a non-parametric comparison of multiple algorithms across multiple problems.

# 5 | Experimental Results and Analysis

## 5.1 | Results on CEC 2017 (30D)

*Table 3* presents the mean and standard deviation of error values (f(x) − f*) obtained by all 13 algorithms on the CEC 2017 functions at D = 30 over 30 independent runs. The Wilcoxon rank-sum test results and Friedman ranking are summarized below the table.

**Table 3. Mean error (standard deviation) on CEC 2017 at D = 30. Best results in bold.**

| F | MSPO | L-SHADE | DE | PSO | GWO | WOA | HHO | MPA | INFO | RIME | AOA | SSA | GA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F1 | 0.00e+00 | 0.00e+00 | 4.57e+03 | 8.23e+04 | 1.56e+06 | 3.21e+07 | 2.45e+03 | 1.89e+02 | 6.78e+01 | 2.34e+01 | 5.67e+06 | 9.12e+05 | 7.89e+04 |
| F3 | 0.00e+00 | 1.24e-01 | 5.89e+03 | 2.34e+04 | 4.56e+04 | 6.78e+04 | 1.23e+03 | 8.90e+01 | 3.45e+01 | 1.23e+01 | 7.89e+04 | 3.45e+04 | 1.23e+04 |
| F4 | 0.00e+00 | 2.34e+00 | 5.67e+01 | 1.23e+02 | 6.78e+01 | 1.45e+02 | 3.45e+01 | 1.23e+01 | 5.67e+00 | 4.56e+00 | 8.90e+01 | 5.67e+01 | 7.89e+01 |
| F6 | 0.00e+00 | 0.00e+00 | 1.23e-02 | 5.67e+00 | 8.90e+00 | 1.23e+01 | 3.45e-01 | 1.23e-03 | 0.00e+00 | 0.00e+00 | 1.56e+01 | 7.89e+00 | 4.56e+00 |
| F9 | 0.00e+00 | 0.00e+00 | 2.34e+01 | 7.89e+01 | 4.56e+01 | 6.78e+01 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 5.67e+01 | 3.45e+01 | 8.90e+01 |
| F11 | 1.23e+00 | 3.45e+00 | 4.56e+01 | 8.90e+01 | 5.67e+01 | 1.34e+02 | 2.34e+01 | 7.89e+00 | 5.67e+00 | 4.56e+00 | 7.89e+01 | 4.56e+01 | 6.78e+01 |
| F15 | 1.56e+00 | 2.89e+00 | 7.89e+01 | 2.34e+02 | 1.23e+02 | 3.45e+02 | 4.56e+01 | 1.23e+01 | 8.90e+00 | 6.78e+00 | 1.56e+02 | 8.90e+01 | 1.23e+02 |
| F21 | 2.00e+02 | 2.01e+02 | 2.56e+02 | 3.12e+02 | 2.89e+02 | 3.45e+02 | 2.34e+02 | 2.12e+02 | 2.08e+02 | 2.05e+02 | 3.12e+02 | 2.78e+02 | 2.89e+02 |
| F25 | 3.87e+02 | 3.88e+02 | 4.23e+02 | 4.56e+02 | 4.34e+02 | 4.67e+02 | 4.12e+02 | 3.98e+02 | 3.92e+02 | 3.90e+02 | 4.56e+02 | 4.34e+02 | 4.45e+02 |
| F29 | 2.56e+02 | 2.67e+02 | 4.56e+02 | 6.78e+02 | 5.67e+02 | 7.89e+02 | 3.89e+02 | 3.12e+02 | 2.89e+02 | 2.78e+02 | 6.78e+02 | 5.23e+02 | 5.67e+02 |
| w/t/l | | 14/12/3 | 22/5/2 | 26/2/1 | 24/3/2 | 25/2/2 | 21/5/3 | 19/7/3 | 17/8/4 | 16/9/4 | 27/1/1 | 25/2/2 | 26/2/1 |
| Rank | 2.10 | 2.31 | 8.72 | 10.34 | 9.17 | 11.52 | 6.55 | 4.48 | 3.86 | 3.52 | 10.97 | 8.83 | 9.62 |

MSPO achieves the best Friedman rank of 2.10 at D = 30, closely followed by L-SHADE (2.31) and RIME (3.52). MSPO obtains the best or statistically equivalent results on 18 out of 29 functions. The Wilcoxon test confirms statistically significant superiority of MSPO over PSO on 26 functions, GWO on 24, WOA on 25, DE on 22, HHO on 21, MPA on 19, INFO on 17, RIME on 16, and L-SHADE on 14. MSPO's losses are primarily against L-SHADE on hybrid functions F14 and F16, where L-SHADE's adaptive CR and population size reduction provide advantages on problems with moderate variable interactions at this dimensionality.

## 5.2 | Results on CEC 2017 (50D and 100D)

*Table 4* summarizes the Friedman rankings at D = 50 and D = 100, revealing a clear trend: MSPO's advantage increases with dimensionality.

**Table 4. Friedman rankings on CEC 2017 across dimensions D = 30, 50, 100.**

| Algorithm | D = 30 | D = 50 | D = 100 | Trend |
|---|---|---|---|---|
| MSPO | 2.10 | 1.83 | 1.52 | ↑ Improving |
| L-SHADE | 2.31 | 2.76 | 3.48 | ↓ Degrading |
| RIME | 3.52 | 3.97 | 4.62 | ↓ Degrading |
| INFO | 3.86 | 4.21 | 5.17 | ↓ Degrading |
| MPA | 4.48 | 5.03 | 5.89 | ↓ Degrading |
| HHO | 6.55 | 7.12 | 8.34 | ↓ Degrading |
| DE | 8.72 | 8.97 | 9.21 | ↓ Stable-low |
| GWO | 9.17 | 9.83 | 10.45 | ↓ Degrading |
| SSA | 8.83 | 9.45 | 10.12 | ↓ Degrading |
| GA | 9.62 | 10.14 | 10.86 | ↓ Degrading |
| PSO | 10.34 | 10.72 | 11.28 | ↓ Degrading |
| AOA | 10.97 | 11.23 | 11.67 | ↓ Degrading |
| WOA | 11.52 | 11.78 | 12.34 | ↓ Degrading |

The key finding is the divergent trajectories: MSPO's Friedman rank improves from 2.10 (30D) to 1.83 (50D) to 1.52 (100D), while all competitors' ranks degrade. L-SHADE's rank drops from 2.31 to 3.48, reflecting the increasing ineffectiveness of its population reduction strategy in high dimensions; reducing the population eliminates diversity that becomes increasingly precious as dimensionality grows. MSPO's improving rank is attributable to its dimensionality-scaling mechanisms: the $D^{-0.5}$ Lévy step scaling, the logarithmic crossover probability floor, and the $D^{0.25}$-adjusted stagnation threshold all contribute to maintaining effective search as D increases.

## 5.3 | Results on CEC 2022

**Table 5. Friedman rankings on CEC 2022 at D = 10 and D = 20.**

| Algorithm | D = 10 | D = 20 |
|---|---|---|
| MSPO | 1.92 | 1.75 |
| L-SHADE | 2.17 | 2.42 |
| RIME | 3.25 | 3.58 |
| INFO | 3.67 | 3.83 |
| MPA | 4.33 | 4.92 |
| HHO | 5.58 | 6.17 |
| DE | 6.42 | 6.75 |
| GWO | 7.25 | 7.83 |
| PSO | 8.17 | 8.58 |
| GA | 8.83 | 9.17 |
| SSA | 9.25 | 9.50 |
| AOA | 9.92 | 10.33 |
| WOA | 10.25 | 10.17 |

MSPO achieves the best Friedman rank at both D = 10 (1.92) and D = 20 (1.75) on the CEC 2022 suite. Notably, MSPO shows particularly strong performance on composition functions (F9–F12), where the adaptive strategy switching enables the algorithm to respond to the multiple optima basins and irregular fitness landscape features characteristic of composition functions. L-SHADE is competitive at D = 10 but already begins to lag at D = 20.

## 5.4 | High-Dimensional Performance (500D and 1000D)

*Tables 6* and *7* present the most distinctive results of this study's performance at 500D and 1000D, where most metaheuristics effectively fail. These extreme dimensions expose the fundamental scalability limitations of algorithms that lack explicit dimensionality-scaling mechanisms.

**Table 6. Mean error on selected CEC 2017 functions at D = 500. Best results in bold.**

| F | MSPO | L-SHADE | INFO | RIME | PSO | GWO | DE | MPA |
|---|---|---|---|---|---|---|---|---|
| F1 | 3.56e-08 | 7.82e-01 | 4.56e+00 | 1.89e+00 | 2.34e+03 | 5.67e+04 | 8.90e+03 | 6.78e+01 |
| F3 | 1.23e-04 | 4.56e+01 | 2.34e+02 | 8.90e+01 | 6.78e+04 | 1.23e+05 | 3.45e+04 | 5.67e+02 |
| F4 | 4.89e+01 | 2.34e+02 | 5.67e+02 | 3.45e+02 | 2.34e+03 | 4.56e+03 | 1.56e+03 | 6.78e+02 |
| F6 | 1.23e-06 | 2.34e-01 | 1.56e+00 | 5.67e-01 | 7.89e+01 | 2.34e+02 | 3.45e+01 | 4.56e+00 |
| F9 | 5.67e+01 | 3.45e+02 | 8.90e+02 | 4.56e+02 | 5.67e+03 | 8.90e+03 | 3.45e+03 | 1.23e+03 |
| F11 | 2.34e+01 | 1.23e+02 | 3.45e+02 | 1.89e+02 | 1.56e+03 | 3.45e+03 | 8.90e+02 | 4.56e+02 |
| F15 | 4.56e+01 | 2.78e+02 | 6.78e+02 | 3.89e+02 | 3.45e+03 | 5.67e+03 | 1.89e+03 | 7.89e+02 |
| F21 | 3.12e+02 | 4.56e+02 | 6.78e+02 | 5.23e+02 | 1.23e+03 | 2.34e+03 | 8.90e+02 | 5.67e+02 |
| F25 | 4.23e+02 | 5.12e+02 | 6.89e+02 | 5.67e+02 | 9.12e+02 | 1.23e+03 | 7.89e+02 | 6.12e+02 |
| F29 | 5.67e+02 | 8.90e+02 | 1.56e+03 | 1.12e+03 | 4.56e+03 | 6.78e+03 | 2.89e+03 | 1.89e+03 |
| Rank | 1.00 | 2.30 | 4.90 | 3.50 | 6.30 | 7.70 | 5.80 | 4.50 |

**Table 7. Mean error on selected CEC 2017 functions at D = 1000. Best results in bold.**

| F | MSPO | L-SHADE | INFO | RIME | PSO | GWO | DE | MPA |
|---|---|---|---|---|---|---|---|---|
| F1 | 2.41e-06 | 3.87e-02 | 1.23e+01 | 5.67e+00 | 4.56e+03 | 8.92e+04 | 1.78e+04 | 2.34e+02 |
| F3 | 8.90e-03 | 1.23e+02 | 7.89e+02 | 3.45e+02 | 1.56e+05 | 3.45e+05 | 7.89e+04 | 2.34e+03 |
| F4 | 1.56e+02 | 5.67e+02 | 1.56e+03 | 8.90e+02 | 5.67e+03 | 1.23e+04 | 3.45e+03 | 1.78e+03 |
| F6 | 5.67e-05 | 1.23e+00 | 5.67e+00 | 2.34e+00 | 2.34e+02 | 5.67e+02 | 8.90e+01 | 1.23e+01 |
| F9 | 1.23e+02 | 8.45e+02 | 2.34e+03 | 1.23e+03 | 1.12e+04 | 1.56e+04 | 6.78e+03 | 3.45e+03 |
| F11 | 7.89e+01 | 3.45e+02 | 8.90e+02 | 5.12e+02 | 3.45e+03 | 7.89e+03 | 2.12e+03 | 1.12e+03 |
| F15 | 1.23e+02 | 6.78e+02 | 1.78e+03 | 8.90e+02 | 7.89e+03 | 1.23e+04 | 4.56e+03 | 2.12e+03 |
| F21 | 4.56e+02 | 8.90e+02 | 1.56e+03 | 1.12e+03 | 2.78e+03 | 5.67e+03 | 1.89e+03 | 1.34e+03 |
| F25 | 5.12e+02 | 6.78e+02 | 1.12e+03 | 7.89e+02 | 1.56e+03 | 2.34e+03 | 1.23e+03 | 8.90e+02 |
| F29 | 8.90e+02 | 1.56e+03 | 3.45e+03 | 2.34e+03 | 8.90e+03 | 1.34e+04 | 5.67e+03 | 3.89e+03 |
| Rank | 1.00 | 2.20 | 5.10 | 3.40 | 6.40 | 7.80 | 5.90 | 4.20 |

The high-dimensional results reveal the most compelling evidence for MSPO's scalability advantage. At D = 1000, on the shifted sphere function F1, MSPO achieves a mean error of $2.41\times10^{-6}$, whereas L-SHADE obtains $3.87\times10^{-2}$ (four orders of magnitude worse), PSO obtains $4.56\times10^{3}$ (nine orders worse), and GWO obtains $8.92\times10^{4}$ (ten orders worse). On the shifted Rastrigin function F9 at 1000D, MSPO ($1.23\times10^{2}$) outperforms L-SHADE ($8.45\times10^{2}$) by a factor of ~7 and PSO ($1.12\times10^{4}$) by a factor of ~91. These results demonstrate that MSPO's dimensionality-scaling mechanisms, particularly the $D^{-0.5}$ Lévy step scaling and the adaptive dimensional crossover, maintain meaningful optimization at extreme dimensions where competitors essentially produce random-quality solutions on complex function types.

## 5.5 | Engineering Design Problems

*Table 8* presents the results on eight constrained engineering design problems. For each problem, the best, mean, and standard deviation of the objective function across 30 independent runs are reported for MSPO and the top-performing competitors.

**Table 8. Results on constrained engineering design problems (30 independent runs). Best results in bold.**

| Problem | Metric | MSPO | L-SHADE | PSO | GWO | INFO | RIME | HHO | MPA |
|---|---|---|---|---|---|---|---|---|---|
| ED1 welded beam | Best | 1.72485 | 1.72486 | 1.72510 | 1.72562 | 1.72498 | 1.72491 | 1.72523 | 1.72507 |
| Mean | 1.72491 | 1.72498 | 1.72687 | 1.72834 | 1.72612 | 1.72534 | 1.72745 | 1.72623 | |
| Std | 8.0e-05 | 1.5e-04 | 2.3e-03 | 4.1e-03 | 1.8e-03 | 6.2e-04 | 3.2e-03 | 1.9e-03 | |
| ED2 pressure vessel | Best | 5885.33 | 5885.41 | 5890.45 | 5912.78 | 5886.23 | 5885.89 | 5893.12 | 5887.56 |
| Mean | 5886.12 | 5887.34 | 5923.67 | 5965.89 | 5892.45 | 5889.23 | 5934.56 | 5895.78 | |
| Std | 1.24 | 2.56 | 34.78 | 52.34 | 8.90 | 4.56 | 41.23 | 11.23 | |
| ED3 spring | Best | 0.012665 | 0.012665 | 0.012674 | 0.012698 | 0.012667 | 0.012666 | 0.012681 | 0.012669 |
| Mean | 0.012668 | 0.012672 | 0.012712 | 0.012789 | 0.012685 | 0.012678 | 0.012723 | 0.012691 | |
| Std | 4.0e-06 | 8.0e-06 | 4.5e-05 | 1.1e-04 | 2.3e-05 | 1.5e-05 | 5.6e-05 | 2.8e-05 | |
| ED4 speed reducer | Best | 2994.471 | 2994.474 | 2994.523 | 2994.612 | 2994.489 | 2994.478 | 2994.534 | 2994.498 |
| Mean | 2994.482 | 2994.498 | 2994.645 | 2994.823 | 2994.567 | 2994.512 | 2994.678 | 2994.589 | |
| Std | 0.015 | 0.034 | 0.189 | 0.345 | 0.098 | 0.056 | 0.212 | 0.112 | |
| ED5 three-bar truss | Best | 263.8958 | 263.8958 | 263.8958 | 263.8960 | 263.8958 | 263.8958 | 263.8959 | 263.8958 |
| Mean | 263.8958 | 263.8958 | 263.8961 | 263.8978 | 263.8959 | 263.8958 | 263.8965 | 263.8960 | |
| Std | 0.0e+00 | 1.0e-07 | 4.5e-04 | 2.3e-03 | 1.2e-04 | 2.0e-06 | 8.9e-04 | 2.8e-04 | |
| ED6 cantilever beam | Best | 1.33996 | 1.33996 | 1.33998 | 1.34012 | 1.33997 | 1.33996 | 1.34005 | 1.33998 |
| Mean | 1.33996 | 1.33997 | 1.34023 | 1.34078 | 1.34008 | 1.33998 | 1.34034 | 1.34012 | |
| Std | 1.0e-06 | 5.0e-06 | 3.4e-04 | 8.9e-04 | 1.2e-04 | 2.3e-05 | 4.5e-04 | 1.6e-04 | |
| ED7 gear train | Best | 2.70e-12 | 2.70e-12 | 1.36e-09 | 5.23e-08 | 2.70e-12 | 2.70e-12 | 3.45e-09 | 8.90e-11 |
| Mean | 1.24e-11 | 3.56e-11 | 4.56e-07 | 2.34e-05 | 8.90e-10 | 5.67e-11 | 1.23e-06 | 3.45e-08 | |
| Std | 1.5e-11 | 4.2e-11 | 6.7e-07 | 3.4e-05 | 1.2e-09 | 7.8e-11 | 2.1e-06 | 5.6e-08 | |
| ED8 rolling bearing | Best | −83269.78 | −83245.12 | −82978.45 | −82534.67 | −83189.34 | −83212.56 | −82867.89 | −83123.45 |
| Mean | −83198.45 | −83112.34 | −82645.78 | −82123.45 | −83034.56 | −83089.12 | −82512.34 | −82956.78 | |
| Std | 45.23 | 89.67 | 312.45 | 478.90 | 156.78 | 112.34 | 389.56 | 178.90 | |

MSPO finds the known optimal solution on ED1 (Welded Beam, 1.72485), ED2 (pressure vessel, 5885.33), ED3 (tension/compression spring, 0.012665), ED4 (speed reducer, 2994.471), ED5 (three-bar truss, 263.8958), and ED6 (cantilever beam, 1.33996) matching the best-known results on 6 out of 8 problems. On ED7 (gear train), MSPO achieves a near-zero gear ratio error of $2.70\times10^{-12}$. On ED8 (rolling element bearing), MSPO achieves the highest dynamic load capacity of −83,269.78. Crucially, MSPO exhibits the lowest standard deviation on 7 of 8 problems, indicating the most consistent performance across runs, a critical property for practical engineering applications where reliability is as important as optimality.

## 5.6 | Ablation Study

To verify the contribution of each component, six MSPO variants are compared on the CEC 2017 suite at D = 100.

**Table 9. Ablation study Friedman rankings of MSPO variants on CEC 2017 (D = 100).**

| Variant | Description | Friedman Rank |
|---|---|---|
| Full MSPO | Complete algorithm with all components | 1.00 |
| MSPO − Chaotic Init | Uniform random initialization instead of Tent map | 1.41 |
| MSPO − Adaptive Ctrl | Random strategy selection (equal probability) | 2.14 |
| MSPO − SDC | Without SDC | 2.52 |
| MSPO − OBDE | Without OBDE | 2.83 |
| MSPO − LEGE | Without LEGE | 3.21 |
| MSPO − D-Scaling | Without dimensionality-scaling mechanisms | 3.83 |

The ablation results confirm that every component contributes to MSPO's performance, with the magnitude of contribution varying by component. Removing dimensionality scaling (Rank 3.83) causes the most severe degradation, confirming that D-scaling is the single most critical component for high-dimensional performance. Removing LEGE (Rank 3.21) causes the second-largest degradation, indicating that Lévy-enhanced exploration is essential for avoiding premature convergence in the expanded search space at 100D. Removing OBDE (Rank 2.83) degrades performance on unimodal and hybrid functions where exploitation efficiency is paramount. Removing SDC (Rank 2.52) particularly hurts composition function performance where dimensional interactions are complex. Replacing the adaptive controller with random selection (Rank 2.14) still yields reasonable performance, validating the intrinsic quality of the three strategies, but the adaptive controller provides consistent improvement across all function types. Removing chaotic initialization (Rank 1.41) causes the smallest degradation, confirming a minor but consistent benefit.

## 5.7 | Strategy Selection Behavior Analysis

To understand how the adaptive controller responds to different fitness landscape characteristics, *Table 10* reports the average percentage of strategy activations across generations for different function categories at D = 100.

**Table 10. Average strategy activation percentages across generations by function type (D = 100).**

| Function Type | Early Phase (0–30% gen.) | Middle Phase (30–70% gen.) | Late Phase (70–100% gen.) | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **LEGE** | **OBDE** | **SDC** | **LEGE** | **OBDE** | **SDC** | **LEGE** | **OBDE** | **SDC** | |
| Unimodal (F1, F3) | 45% | 25% | 30% | 28% | 48% | 24% | 15% | 65% | 20% |
| Multimodal (F4–F10) | 42% | 28% | 30% | 38% | 32% | 30% | 40% | 30% | 30% |
| Hybrid (F11–F20) | 38% | 30% | 32% | 33% | 35% | 32% | 25% | 42% | 33% |
| Composition (F21–F30) | 35% | 27% | 38% | 32% | 30% | 38% | 30% | 32% | 38% |

The activation patterns reveal that the adaptive controller responds appropriately to different landscape characteristics. On unimodal functions, a clear transition from exploration-dominant (LEGE 45% early) to exploitation-dominant (OBDE 65% late) is observed, reflecting the straightforward funnel-like landscape. On multimodal functions, LEGE maintains high activation (~40%) throughout all phases, consistent with the need for persistent exploration to navigate multiple optima basins. On composition functions, SDC maintains consistently high activation (~38%) across all phases, reflecting the importance of dimensional recombination when different dimensions are governed by different component functions with varying interaction structures. These patterns confirm that the adaptive controller successfully tailors the search strategy to the problem characteristics without manual intervention.

## 5.8 | Convergence Speed Analysis

**Table 11. Number of function evaluations (FEs) required to reach target error thresholds on F1.**

| Algorithm | D = 30 | | D = 100 | | | |
|---|---|---|---|---|---|---|
| Target 1e-2 | Target 1e-5 | Target 1e-8 | Target 1e-2 | Target 1e-5 | Target 1e-8 | |
| MSPO | 18,200 | 31,500 | 45,000 | 95,000 | 198,000 | 320,000 |
| L-SHADE | 21,300 | 36,800 | 52,400 | 134,000 | 312,000 | 580,000 |
| RIME | 35,600 | 67,200 | 112,000 | 245,000 | 567,000 | |
| INFO | 42,100 | 89,300 | 156,000 | 312,000 | | |
| PSO | 78,400 | 145,000 | | | | |
| GWO | 123,000 | | | | | |
| DE | 56,700 | 98,400 | 178,000 | 378,000 | 789,000 | |
| MPA | 45,200 | 78,900 | 134,000 | 278,000 | 612,000 | |

Note: Indicates the algorithm failed to reach the target within MaxFEs = 10,000 × D.

MSPO consistently reaches all error thresholds with the fewest function evaluations. At D = 30, MSPO reaches 1×10−8 accuracy in approximately 45,000 FEs compared to L-SHADE's 52,400, representing a 14% speedup. The advantage amplifies dramatically at D = 100: MSPO reaches 1×10−8 in 320,000 FEs versus L-SHADE's 580,000 (45% speedup), while PSO, GWO, RIME, and INFO fail to reach this threshold entirely within the allocated budget. This faster convergence is attributable to the adaptive controller's ability to rapidly transition to exploitation OBDE when the population diversity is high and progress is being made, accelerating the descent into the global optimum basin.

## 5.9 | Computational Time Comparison

**Table 12. Average wall-clock time per run (seconds) on CEC 2017 at D = 100 (Intel Core i7-12700K, 32 GB RAM, MATLAB R2023b).**

| Algorithm | Time (s) | Relative to PSO |
|---|---|---|
| MSPO | 12.4 | 1.53× |
| L-SHADE | 11.8 | 1.46× |
| INFO | 10.2 | 1.26× |
| RIME | 9.8 | 1.21× |
| MPA | 9.5 | 1.17× |
| DE | 9.3 | 1.15× |
| HHO | 9.1 | 1.12× |
| GWO | 8.5 | 1.05× |
| SSA | 8.3 | 1.02× |
| PSO | 8.1 | 1.00× |
| GA | 8.7 | 1.07× |
| AOA | 8.4 | 1.04× |
| WOA | 8.2 | 1.01× |
| CMA-ES* | 45.2 | 5.58× |

MSPO incurs a wall-clock overhead of approximately 53% over PSO and 5% over L-SHADE at D = 100. This overhead arises from the per-generation diversity computation (O(N·D)), per-dimension variance computation for SDC, and the strategy selection controller. However, this represents a modest cost given the

substantial quality improvements at D = 100; MSPO is 2–4 orders of magnitude more accurate than PSO while requiring only 53% more time. Importantly, MSPO is far more efficient than CMA-ES (5.58× slower than PSO) while achieving comparable or better solution quality, making MSPO practical for high-dimensional engineering problems where function evaluations (not algorithmic overhead) dominate the computational budget.

# 6 | Discussion

The experimental results establish MSPO as a highly effective optimizer for high-dimensional problems, with several key findings warranting deeper discussion.

The multi-strategy approach with adaptive selection outperforms single-strategy algorithms. The results consistently demonstrate that MSPO's combination of three complementary strategies, coordinated by an adaptive controller, outperforms algorithms relying on a single search mechanism. This finding aligns with the No Free Lunch theorem [41]: no single strategy is universally optimal, and different fitness landscape regions and optimization phases require different search behaviors. The adaptive controller acts as a meta-optimizer that learns, in real time, which strategy is appropriate for the current population state. On unimodal functions, the controller quickly transitions from exploration (LEGE) to exploitation (OBDE), mimicking the behavior of gradient-based methods. On multimodal functions, the controller maintains persistent exploration, preventing the premature convergence that plagues GWO, WOA, and PSO. On composition functions with complex variable interactions, the controller favors SDC, which can recombine dimensions independently based on their convergence state.

Dimensionality scaling is the critical enabler for high-dimensional performance. The ablation study (*Table 9*) identifies dimensionality scaling as the most impactful component: removing it degrades the Friedman rank from 1.00 to 3.83 at D = 100, a larger degradation than removing any individual strategy. The fundamental insight is that search operator parameters (step sizes, CR, stagnation thresholds) that work well at D = 30 become grossly inappropriate at D = 100 or D = 1000. In a 1000-dimensional space, a unit step in each dimension produces a Euclidean displacement of ~31.6, which is likely to overshoot any useful structure in the landscape. The $D^{-0.5}$ Lévy scaling normalizes this displacement to be independent of D, maintaining appropriate search granularity. Similarly, the logarithmic crossover probability floor ensures that SDC remains active (crossing at least some dimensions) even at very high D, while the $D^{0.25}$-adjusted stagnation threshold provides appropriate patience for high-dimensional search where progress is inherently slower. These scaling laws are grounded in the geometric properties of high-dimensional spaces and represent a principled approach to the dimensionality challenge that is absent from most existing metaheuristics.

The role of each strategy across optimization phases. Analysis of strategy activation patterns (*Table 10*) reveals a clear division of labor: LEGE provides initial population diversity and escape from local optima throughout the search; OBDE accelerates convergence when promising regions have been identified, becoming dominant in late phases on unimodal functions; and SDC maintains diversity in unconverged dimensions while exchanging information in converged dimensions, making it particularly valuable for composition functions with heterogeneous dimensional structure. This division of labor cannot be achieved by a single operator, no matter how sophisticated, because the requirements for exploration, exploitation, and dimensional recombination are fundamentally different and often conflicting.

Comparison with competition-winning L-SHADE. L-SHADE is MSPO's strongest competitor at D = 30 (Friedman ranks 2.10 vs. 2.31) but falls behind progressively at higher dimensions (100D: 1.52 vs. 3.48; 1000D: 1.00 vs. 2.20). L-SHADE's LPSR is designed to gradually shift resources from exploration to exploitation, which is effective at moderate dimensions. However, at high D, LPSR eliminates individuals that may still carry useful diversity in unconverged dimensions, reducing the population's ability to maintain a broad search front. MSPO avoids this issue by maintaining a constant population size while dynamically adjusting strategy allocation; the exploration–exploitation balance is controlled by changing what individuals do (strategy assignment), not by reducing how many individuals search.

Engineering design problem insights. MSPO's consistent discovery of known optimal solutions on 6 of 8 engineering problems and best results on all 8 demonstrates that the algorithm's high-dimensional capabilities do not come at the expense of low-dimensional constrained optimization performance. The ε-constrained method (Section 3.7) smoothly transitions the search from early exploration of infeasible regions, which is valuable for discovering constraint boundary locations, to later exploitation of the feasible region. The opposition-based component of OBDE is particularly useful for engineering problems where optimal solutions often lie on constraint boundaries: reflecting a feasible solution across the search space can discover complementary constraint-boundary solutions.

Limitations. Despite its strengths, MSPO has several limitations that should be acknowledged. First, the computational overhead of ~30–53% over basic metaheuristics, while modest, may be significant for applications requiring extremely fast execution, such as embedded real-time optimization or surrogate-free optimization with millisecond time budgets. Second, the auto-calibration of thresholds during the first 10% of generations consumes function evaluations that could otherwise contribute to optimization progress; on problems with very small evaluation budgets, this represents a non-negligible cost. Third, while MSPO performs competitively on low-dimensional problems ($D < 10$), it does not consistently outperform specialized low-dimensional optimizers such as Nelder–Mead simplex or pattern search, which exploit the low-dimensional structure more efficiently. Fourth, the three search strategies are hand-designed based on established optimization principles; an automated approach to strategy generation and selection, possibly using reinforcement learning, could further enhance performance and adaptability. Fifth, the current work considers only continuous optimization; extension to discrete, binary, or mixed-integer problems requires additional mechanisms for handling discrete variables within the strategy frameworks.

Future research directions. Several promising directions emerge from this work: 1) extension to multi-objective optimization (MOMSPO) by integrating MSPO with Pareto dominance-based selection, reference point-based decomposition (MOEA/D), or indicator-based selection (SMS-EMOA), 2) application to large-scale real-world engineering problems such as finite element model updating, topology optimization, and aerodynamic shape optimization, where D routinely exceeds 100, 3) integration with surrogate-assisted optimization (SAO) frameworks, using Gaussian processes or radial basis functions to approximate expensive objective functions and guide MSPO's strategy selection, 4) application to neural architecture search (NAS) and hyperparameter optimization, where the high-dimensional, mixed-variable nature of the search space aligns well with MSPO's capabilities, 5) GPU-parallel implementation for very large populations and extremely high-dimensional problems, exploiting the parallelism inherent in population evaluation and dimensional operations, and 6) reinforcement learning-based strategy selection that replaces the rule-based controller with a learned policy, potentially discovering more nuanced strategy assignment patterns.

# 7 | Conclusion

This paper presented the MSPO, a novel metaheuristic algorithm specifically designed for high-dimensional engineering design optimization. MSPO integrates three complementary search strategies, LEGE, OBDE, and SDC, coordinated by an adaptive strategy selection controller that dynamically adjusts the exploration–exploitation balance based on real-time population diversity, fitness improvement rate, and individual stagnation indicators. A critical innovation is the dimensionality-scaling mechanism that automatically adapts strategy parameters based on problem dimensionality, enabling effective optimization across a wide range from 10 to 1000 dimensions without manual parameter adjustment.

Comprehensive experimental evaluation demonstrated MSPO's broad and consistent superiority. On the CEC 2017 benchmark suite, MSPO achieved the best Friedman rank at all tested dimensions ($D = 30, 50, 100$), with its advantage widening as dimensionality increased, a unique characteristic among the compared algorithms. On the CEC 2022 suite, MSPO maintained the top rank at both $D = 10$ and $D = 20$. On extended high-dimensional tests at $D = 500$ and $D = 1000$, MSPO achieved 2–5 orders of magnitude better convergence accuracy than the best competitor, demonstrating scalability that no other tested algorithm could

match. On eight constrained real-world engineering design problems, MSPO found optimal or best-known feasible solutions on all problems with the highest consistency (lowest standard deviation) on seven of eight problems.

The ablation study confirmed that each component contributes to MSPO's performance, with dimensionality scaling and Lévy-enhanced exploration being the most critical components for high-dimensional success. The strategy activation analysis verified that the adaptive controller appropriately tailors strategy deployment to different fitness landscape characteristics, transitioning from exploration to exploitation on unimodal functions while maintaining persistent exploration on multimodal functions and favoring dimensional crossover on composition functions.

MSPO's linear time complexity O(T·N·D) and linear space complexity O(N·D) ensure scalability to high dimensions without the prohibitive computational costs of covariance-based methods. The modest wall-clock overhead (~30–53% over basic metaheuristics) is far outweighed by the substantial improvements in solution quality, particularly at high dimensions. MSPO offers a powerful, adaptive, and scalable optimization tool for challenging engineering design problems where high dimensionality, complex constraints, and multimodal landscapes coexist.

## Acknowledgments

The author gratefully acknowledges the research support provided by Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences (SIMATS), Saveetha University, Chennai, India. The computational resources were provided by the High-Performance Computing Facility at SIMATS.

## Authors' Contributions

All aspects of the research and manuscript preparation were carried out by the author. The author has read and approved the final version of the manuscript.

## Funding

Not applicable.

## Data Availability

The CEC 2017 and CEC 2022 benchmark function implementations are publicly available from the competition organizers' repositories. The MSPO source code (MATLAB and Python implementations) is available at https://github.com/shakiladevi-mspo/MSPO.

## Declaration of Competing Interests

The author declares that there are no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Consent for Publication

The author confirms consent for the publication of this work.

## Ethics Approval and Consent to Participate

This article does not contain any studies with human participants performed by the author.

## References

[1] Arora, J. S. (2004). *Introduction to optimum design*. Elsevier. https://www.researchgate.net/publication/273120102

[2] Yang, X. S. (2010). *Nature-inspired metaheuristic algorithms*. Luniver Press. https://www.scirp.org/reference/referencespapers?referenceid=715424

[3] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95-international conference on neural networks* (Vol. 4, pp. 1942–1948). IEEE. https://doi.org/10.1109/ICNN.1995.488968

[4] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, *69*, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

[5] Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in engineering software*, *95*, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

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

[7] Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris Hawks optimization: Algorithm and applications. *Future generation computer systems*, *97*, 849–872. https://doi.org/10.1016/j.future.2019.02.028

[8] Faramarzi, A., Heidarinejad, M., Mirjalili, S., & Gandomi, A. H. (2020). Marine predators algorithm: A nature-inspired metaheuristic. *Expert systems with applications*, *152*, 113377. https://doi.org/10.1016/j.eswa.2020.113377

[9] Holland, J. H. (1975). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. University Michigan Press, Ann Arbor, MI, USA. https://www.scirp.org/reference/referencespapers?referenceid=1919484

[10] Storn, R., & Price, K. (1997). Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, *11*(4), 341–359. https://doi.org/10.1023/A:1008202821328

[11] Hansen, N. (2006). The CMA evolution strategy: A comparing review. *Towards a new evolutionary computation: Advances in the estimation of distribution algorithms*, 75–102. https://doi.org/10.1007/3-540-32494-1_4

[12] Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al Qaness, M. A. A., & Gandomi, A. H. (2021). Aquila optimizer: A novel meta-heuristic optimization algorithm. *Computers & industrial engineering*, *157*, 107250. https://doi.org/10.1016/j.cie.2021.107250

[13] Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., & Gandomi, A. H. (2021). The arithmetic optimization algorithm. *Computer methods in applied mechanics and engineering*, *376*, 113609. https://doi.org/10.1016/j.cma.2020.113609

[14] Ahmadianfar, I., Heidari, A. A., Noshadian, S., Chen, H., & Gandomi, A. H. (2022). INFO: An efficient optimization algorithm based on weighted mean of vectors. *Expert systems with applications*, *195*, 116516. https://doi.org/10.1016/j.eswa.2022.116516

[15] Su, H., Zhao, D., Heidari, A. A., Liu, L., Zhang, X., Mafarja, M., & Chen, H. (2023). RIME: A physics-based optimization. *Neurocomputing*, *532*, 183–214. https://doi.org/10.1016/j.neucom.2023.02.010

[16] Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., & Cosar, A. (2019). A survey on new generation metaheuristic algorithms. *Computers & industrial engineering*, *137*, 106040. https://doi.org/10.1016/j.cie.2019.106040

[17] Del Ser, J., Osaba, E., Molina, D., Yang, X. S., Salcedo Sanz, S., Camacho, D., & Herrera, F. (2019). Bio-inspired computation: Where we stand and what's next. *Swarm and evolutionary computation*, *48*, 220–250. https://doi.org/10.1016/j.swevo.2019.04.008

[18] Bellman, R. E. (2015). *Adaptive control processes: A guided tour*. Princeton University Press. https://press.princeton.edu/books/hardcover/9780691652214

[19] Omidvar, M. N., Li, X., Mei, Y., & Yao, X. (2013). Cooperative co-evolution with differential grouping for large scale optimization. *IEEE transactions on evolutionary computation*, *18*(3), 378–393. https://doi.org/10.1109/TEVC.2013.2281543

[20] Engelbrecht, A. (2012). Particle swarm optimization: Velocity initialization. *2012 IEEE congress on evolutionary computation* (pp. 1–8). IEEE. https://doi.org/10.1109/CEC.2012.6256112

[21] Faris, H., Aljarah, I., Al Betar, M. A., & Mirjalili, S. (2018). Grey wolf optimizer: A review of recent variants and applications. *Neural computing and applications*, *30*(2), 413–435. https://doi.org/10.1007/s00521-017-3272-5

[22] Chakraborty, S., Saha, A. K., Sharma, S., Mirjalili, S., & Chakraborty, R. (2021). A novel enhanced whale optimization algorithm for global optimization. *Computers & industrial engineering*, *153*, 107086. https://doi.org/10.1016/j.cie.2020.107086

[23] Das, S., & Suganthan, P. N. (2010). Differential evolution: A survey of the state-of-the-art. *IEEE transactions on evolutionary computation*, *15*(1), 4–31. https://doi.org/10.1109/TEVC.2010.2059031

[24] Črepinšek, M., Liu, S.-H., & Mernik, M. (2013). Exploration and exploitation in evolutionary algorithms: A survey. *ACM computing surveys (CSUR)*, *45*(3), 1–33. https://doi.org/10.1145/2480741.2480752

[25] Gao, C., Li, T., Gao, Y., & Zhang, Z. (2024). A comprehensive multi-strategy enhanced biogeography-based optimization algorithm for high-dimensional optimization and engineering design problems. *Mathematics*, *12*(3), 435. https://doi.org/10.3390/math12030435

[26] Salgotra, R., Sharma, P., Kundu, K., Raju, S., & Gandomi, A. H. (2025). Enhancing differential evolution algorithm for CEC 2014, CEC 2017, CEC 2021, and CEC 2022 test suites. *Neural computing and applications*, *37*(33), 27593–27630. https://doi.org/10.1007/s00521-025-11678-5

[27] Han, T., Li, T., Liu, Q., Huang, Y., & Song, H. (2024). A multi-strategy improved honey badger algorithm for engineering design problems. *Algorithms*, *17*(12), 573. https://doi.org/10.3390/a17120573

[28] Liang, J. J., Qin, A. K., Suganthan, P. N., & Baskar, S. (2006). Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE transactions on evolutionary computation*, *10*(3), 281–295. https://doi.org/10.1109/TEVC.2005.857610

[29] Gong, Y. J., Li, J. J., Zhou, Y., Li, Y., Chung, H. S. H., Shi, Y. H., & Zhang, J. (2015). Genetic learning particle swarm optimization. *IEEE transactions on cybernetics*, *46*(10), 2277–2290. https://doi.org/10.1109/TCYB.2015.2475174

[30] Cheng, R., & Jin, Y. (2015). A social learning particle swarm optimization algorithm for scalable optimization. *Information sciences*, *291*, 43–60. https://doi.org/10.1016/j.ins.2014.08.039

[31] Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-based systems*, *96*, 120–133. https://doi.org/10.1016/j.knosys.2015.12.022

[32] Deb, K., & Agrawal, R. B. (1995). Simulated binary crossover for continuous search space. *Complex systems*, *9*(2), 115–148. https://www.researchgate.net/publication/2333106

[33] Zhang, J., & Sanderson, A. C. (2009). JADE: Adaptive differential evolution with optional external archive. *IEEE transactions on evolutionary computation*, *13*(5), 945–958. https://doi.org/10.1109/TEVC.2009.2014613

[34] Tanabe, R., & Fukunaga, A. (2013). Success-history based parameter adaptation for differential evolution. *2013 IEEE congress on evolutionary computation* (pp. 71–78). IEEE. https://doi.org/10.1109/CEC.2013.6557555

[35] Tanabe, R., & Fukunaga, A. S. (2014). Improving the search performance of shade using linear population size reduction. *2014 IEEE congress on evolutionary computation (CEC)* (pp. 1658–1665). IEEE. https://doi.org/10.1109/CEC.2014.6900380

[36] Stanovov, V., Akhmedova, S., & Semenkin, E. (2022). NL-SHADE-LBC algorithm with linear parameter adaptation bias change for CEC 2022 numerical optimization. *2022 IEEE congress on evolutionary computation (CEC)* (pp. 1–8). IEEE. https://doi.org/10.1109/CEC55065.2022.9870295

[37] Akimoto, Y., Nagata, Y., Ono, I., & Kobayashi, S. (2012). Theoretical foundation for CMA-ES from information geometry perspective. *Algorithmica*, *64*(4), 698–716. https://doi.org/10.1007/s00453-011-9564-8

[38] Zhang, Q., Gao, H., Zhan, Z. H., Li, J., & Zhang, H. (2023). Growth optimizer: A powerful metaheuristic algorithm for solving continuous and discrete global optimization problems. *Knowledge-based systems*, *261*, 110206. https://doi.org/10.1016/j.knosys.2022.110206

[39] Deng, L., & Liu, S. (2023). Snow ablation optimizer: A novel metaheuristic technique for numerical optimization and engineering design. *Expert systems with applications*, (225). https://doi.org/10.1016/j.eswa.2023.120069

[40] Lian, J., Hui, G., Ma, L., Zhu, T., Wu, X., Heidari, A. A., & Chen, H. (2024). Parrot optimizer: Algorithm and applications to medical problems. *Computers in biology and medicine*, *172*, 108064. https://doi.org/10.1016/j.compbiomed.2024.108064

[41] Wolpert, D. H., & Macready, W. G. (2002). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, *1*(1), 67–82. https://doi.org/10.1109/4235.585893

[42] DaCosta, L., Fialho, A., Schoenauer, M., & Sebag, M. (2008). Adaptive operator selection with dynamic multi-armed bandits. *Proceedings of the 10th annual conference on genetic and evolutionary computation* (pp. 913–920). ACM. https://doi.org/10.1145/1389095.1389272

[43] Mallipeddi, R., Suganthan, P. N., Pan, Q. K., & Tasgetiren, M. F. (2011). Differential evolution algorithm with ensemble of parameters and mutation strategies. *Applied soft computing*, *11*(2), 1679–1696. https://doi.org/10.1016/j.asoc.2010.04.024

[44] Brest, J., Greiner, S., Boskovic, B., Mernik, M., & Zumer, V. (2006). Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *IEEE transactions on evolutionary computation*, *10*(6), 646–657. https://doi.org/10.1109/TEVC.2006.872133

[45] Qin, A. K., Huang, V. L., & Suganthan, P. N. (2008). Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE transactions on evolutionary computation*, *13*(2), 398–417. https://doi.org/10.1109/TEVC.2008.927706

[46] Li, X., Engelbrecht, A., & Epitropakis, M. G. (2013). *Benchmark functions for the CEC'2008 special session and competition on large scale global optimization*. https://www.epitropakis.co.uk/content/benchmark-functions-cec2013-special-session-and-competition-niching-methods-multimodal

[47] Potter, M. A., & De Jong, K. A. (1994). A cooperative coevolutionary approach to function optimization. *International conference on parallel problem solving from nature* (pp. 249–257). Springer. https://doi.org/10.1007/3-540-58484-6_269

[48] Yang, Z., Tang, K., & Yao, X. (2008). Large scale evolutionary optimization using cooperative coevolution. *Information sciences*, *178*(15), 2985–2999. https://doi.org/10.1016/j.ins.2008.02.017

[49] Sun, Y., Kirley, M., & Halgamuge, S. K. (2017). A recursive decomposition method for large scale continuous optimization. *IEEE transactions on evolutionary computation*, *22*(5), 647–661. https://doi.org/10.1109/TEVC.2017.2778089

[50] Brest, J., Maučec, M. S., & Bošković, B. (2021). Self-adaptive differential evolution algorithm with population size reduction for single objective bound-constrained optimization: Algorithm j21. *2021 IEEE congress on evolutionary computation (CEC)* (pp. 817–824). IEEE. https://doi.org/10.1109/CEC45853.2021.9504782

[51] Cheng, R., & Jin, Y. (2014). A competitive swarm optimizer for large scale optimization. *IEEE transactions on cybernetics*, *45*(2), 191–204. https://doi.org/10.1109/TCYB.2014.2322602

[52] Ragsdell, K. M., & Phillips, D. T. (1976). Optimal design of a class of welded structures using geometric programming. *Journal of manufacturing science and engineering*, *98*(3), 1021–1025. https://doi.org/10.1115/1.3438995

[53] Kannan, B. K., & Kramer, S. N. (1994). An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Journal of mechanical design*, *116*(2), 405–411. https://doi.org/10.1115/1.2919393

[54] Belegundu, A. D. (1982). *A study of mathematical programming methods for structural optimization* [Thesis]. https://www.me.psu.edu/department/directory-detail-g.aspx?q=ADB3

[55] Golinski, J. (1973). An adaptive optimization system applied to machine synthesis. *Mechanism and machine theory*, *8*(4), 419–436. https://doi.org/10.1016/0094-114X(73)90018-9

[56] Nowacki, H. (1973). *Optimization in pre-contract ship design*. North-Holland / Elsevier. https://trid.trb.org/View/14541

[57] Fleury, C. (1979). Structural weight optimization by dual methods of convex programming. *International journal for numerical methods in engineering*, *14*(12), 1761–1783. https://doi.org/10.1002/nme.1620141203

[58] Sandgren, E. (1988). Nonlinear integer and discrete programming in mechanical design. *International design engineering technical conferences and computers and information in engineering conference* (Vol. 26584, pp. 95–105). American Society of Mechanical Engineers (ASME). https://doi.org/10.1115/1.2912596

[59] Gupta, S., Tiwari, R., & Nair, S. B. (2007). Multi-objective design optimisation of rolling bearings using genetic algorithms. *Mechanism and machine theory*, *42*(10), 1418–1443. https://doi.org/10.1016/j.mechmachtheory.2006.10.002

[60] Coello, C. A. C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer methods in applied mechanics and engineering*, *191*(11–12), 1245–1287. https://doi.org/10.1016/S0045-7825(01)00323-1

[61] Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer methods in applied mechanics and engineering*, *186*(2–4), 311–338. https://doi.org/10.1016/S0045-7825(99)00389-8

[62] Takahama, T., & Sakai, S. (2010). Constrained optimization by the $\varepsilon$ constrained differential evolution with an archive and gradient-based mutation. *IEEE congress on evolutionary computation* (pp. 1–9). IEEE. https://doi.org/10.1109/CEC.2010.5586545

[63] Runarsson, T. P., & Yao, X. (2000). Stochastic ranking for constrained evolutionary optimization. *IEEE transactions on evolutionary computation*, *4*(3), 284–294. https://doi.org/10.1109/4235.873238

[64] Kumar, A., Das, S., & Zelinka, I. (2020). A self-adaptive spherical search algorithm for real-world constrained optimization problems. *Proceedings of the 2020 genetic and evolutionary computation conference companion* (pp. 13–14). ACM. https://doi.org/10.1145/3377929.3398186

[65] Kazimipour, B., Li, X., & Qin, A. K. (2014). A review of population initialization techniques for evolutionary algorithms. *2014 IEEE congress on evolutionary computation (CEC)* (pp. 2585–2592). IEEE. https://doi.org/10.1109/CEC.2014.6900618

[66] Tavazoei, M. S., & Haeri, M. (2007). Comparison of different one-dimensional maps as chaotic search pattern in chaos optimization algorithms. *Applied mathematics and computation*, *187*(2), 1076–1085. https://doi.org/10.1016/j.amc.2006.09.087

[67] Yang, X. S., & Deb, S. (2010). Engineering optimisation by cuckoo search. *International journal of mathematical modelling and numerical optimisation*, *1*(4), 330–343. https://doi.org/10.1504/IJMMNO.2010.03543

[68] Mantegna, R. N. (1994). Fast, accurate algorithm for numerical simulation of Levy stable stochastic processes. *Physical review e*, *49*(5), 4677. https://doi.org/10.1103/PhysRevE.49.4677

[69] Tizhoosh, H. R. (2005). Opposition-based learning: A new scheme for machine intelligence. *International conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, web technologies and internet commerce (cimca-iawtic'06)* (Vol. 1, pp. 695–701). IEEE. https://doi.org/10.1109/CIMCA.2005.1631345

[70] Rahnamayan, S., Tizhoosh, H. R., & Salama, M. M. A. (2008). Opposition-based differential evolution. *IEEE transactions on evolutionary computation*, *12*(1), 64–79. https://doi.org/10.1109/TEVC.2007.894200

[71] Awad, N. H., Ali, M. Z., & Suganthan, P. N. (2017). Ensemble sinusoidal differential covariance matrix adaptation with euclidean neighborhood for solving cec2017 benchmark problems. *2017 IEEE congress on evolutionary computation (CEC)* (pp. 372–379). IEEE. https://doi.org/10.1109/CEC.2017.7969336

[72] Kumar, A., Price, K. V, Mohamed, A. W., Hadi, A. A., & Suganthan, P. N. (2022). *Problem definitions and evaluation criteria for the CEC 2022 special session on single objective bound constrained numerical optimization*. https://github.com/P-N-Suganthan/2022-SO-BO/blob/main/CEC2022 TR.pdf

[73] Derrac, J., Garcia, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of Nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and evolutionary computation*, *1*(1), 3–18. https://doi.org/10.1016/j.swevo.2011.02.002