Social Dissent in Particle Swarm Optimization: Breaking Groupthink via Adaptive Stochastic Noise

Swapnoneel Sarkar
Accelequant India Pvt. Ltd.
Bengaluru, India

Somnath Mukhopadhyay
Dept. of Computer Science and Engineering
Assam University
Silchar 788011, Assam, India

Andries P. Engelbrecht
Dept. of Industrial Engineering and
Computer Science Division
Stellenbosch University, South Africa

***Abstract*—Adaptive social-dissent particle swarm optimization (PSO) addresses the vulnerability of PSO to groupthink and social lock-in by introducing social dissent, which refers to deliberate disagreement in social learning. The framework utilizes three operators: leader corruption, where particles follow foreign leaders; membership migration, where particles temporarily switch to another sub-swarm; and the scout operator, which triggers stochastic re-initialization and momentum resampling for stagnated particles to relocate them to unvisited areas of the search space. Dissent intensity is modulated by the path length ratio, a straightness measure that scales effective probabilities to encourage exploration in oscillatory particles while allowing steadily moving particles to converge. When combined with topology annealing that linearly reduces the number of sub-swarms to balance parallel exploration and unified exploitation, the algorithm achieves superior expected running time and success rates on COCO/BBOB benchmarks. These improvements are particularly pronounced in high-dimensional tasks where the method counteracts diversity collapse and prevents the social lock-in that traps traditional implementations.**

I. Introduction

Particle swarm optimization (PSO) [1] is an effective black-box optimizer, yet it remains vulnerable to groupthink. This occurs when particles over-trust social signals, causing diversity to vanish and leading to premature convergence [2], [3]. On rugged or deceptive landscapes, this often manifests as "social lock-in,": particles remain trapped within sub-optimal basins despite continuous motion. A social-dissent particle swarm optimizer (SD-PSO) is proposed: deliberate, bounded disagreement in social learning is introduced to maintain robustness. Dissent is implemented via leader corruption and membership migration operators that perturb information flow, forcing particles to evaluate positions that would otherwise remain inaccessible under standard social learning.

To further mitigate stagnation, a scout operator facilitates radical diversity through stochastic re-initialization. These scouts follow "unsafe" trajectories that violate traditional stochastic stability bounds [4], enabling the swarm to escape local optima situated beyond the range of restricted velocity clamping. Crucially, dissent intensity is modulated locally by a per-particle path length ratio (PLR), a "straightness" measure ensuring that noise is injected only when progress stagnates. When combined with topology annealing to transition from parallel exploration to unified exploitation [5], diversity collapse common in complex search spaces is addressed.

SD-PSO is evaluated using the COCO/BBOB benchmarks [6], [7] against standard PSO [8] and hybrid variants such as DE-PSO [9]. The results demonstrate superior expected runtime and success rates, particularly on higher-dimensional problems where dissent operators effectively counteract social lock-in. The primary contributions of this work are: (1) a compact dissent mechanism (leader corruption and membership migration) that perturbs social information flow; (2) a per-particle PLR-driven adaptation rule that scales dissent probabilities online; and (3) a comprehensive COCO/BBOB evaluation reporting aggregate summaries and stability-map analysis. The source code for SD-PSO is available at https://github.com/swapsarkar/Social-Dissent_PSO.

The paper is organized as follows: Section II reviews related work, Section III details the SD-PSO operators and annealing, Section IV outlines the experimental setup, Section V presents the results, and Section VI concludes the paper.

II. Related Work

PSO prematurely converges when social learning rapidly collapses swarm diversity. PSO research extensively explored (i) *how* social information is formed and propagated [2], [10], (ii) *when* diversification is triggered via stagnation detectors [4], [11], and (iii) *where* information flows via topology control and multi-swarm designs [12]. Refer to [13] for more detail on PSO variations.

This section reviews prior strategies designed to mitigate premature convergence by perturbing particle search trajectories and by regulating diversity in particle positions and velocities within the swarm. These approaches are typically guided by stagnation detection, characterized by particle velocities approaching or reaching zero, triggering adaptive control mechanisms. Such mechanisms involve injection or reduction of diversity, or deliberate intervention in the search dynamics through modification of particle positions or velocities, either for a subset of particles or the entire swarm, thereby disrupting the swarm's progression and helping it escape local optima.

Section II.A defines the inertia-weight PSO. Section II.B details disruption and diversity modification mechanisms. Section II.C covers adaptive mechanisms and methods to detect stagnation. Section II.D covers methods involving multiple swarms that optimize individually, as well as topological annealing, where swarms are slowly combined during the search process. Section

II.E covers how PSO stability or the tendency of the particles to converge toward a fixed point (i.e., an equilibrium) rather than diverging toward infinity or oscillating uncontrollably is modeled.

A. Inertia-Weight Particle Swarm Optimization

The inertia weight PSO (IW-PSO) was introduced in 1998 [14]. Each particle i maintains a position $\mathbf{x}_i \in \mathbb{R}^d$, a velocity $\mathbf{v}_i \in \mathbb{R}^d$, and a personal best $\mathbf{p}_i \in \mathbb{R}^d$. Social learning uses a leader (neighborhood/global best) signal $\mathbf{n}_i \in \mathbb{R}^d$.

Particle velocity is updated as

$$\begin{aligned} \mathbf{v}_i(t+1) &= \omega\,\mathbf{v}_i(t) + c_1\,\mathbf{r}_{1i}(t) \odot (\mathbf{p}_i(t) - \mathbf{x}_i(t)) \\ &+ c_2\,\mathbf{r}_{2i}(t) \odot (\mathbf{n}_i(t) - \mathbf{x}_i(t)), \end{aligned} \tag{1}$$

where ω is the inertia weight, c_1 and c_2 are the acceleration coefficients, $\mathbf{p}_i$ and $\mathbf{n}_i$ are respectively the personal best and neighborhood best positions, and components of $\mathbf{r}_{1i}, \mathbf{r}_{2i}$ are sampled as $\mathbf{r}_{1ij}, \mathbf{r}_{2ij} \sim \mathrm{Unif}(0,1)$ independently, and $\odot$ denotes element-wise multiplication. Each particle position is updated as $\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1)$. A boundary constraint handling mechanism is applied to ensure feasibility within $[\ell, u]^d$ as $x_{ij}(t{+}1) \leftarrow \min\big(\max(x_{ij}(t{+}1), \ell_j), u_j\big),\ j = 1, \ldots, d$.

B. Disruption and Diversity Mechanisms

A foundational line of work focuses on stable swarm dynamics. The constriction-factor analysis of Clerc and Kennedy provides sufficient conditions for bounded trajectories under canonical PSO updates [4]. Richer informer models broaden social influence. In fully informed PSO (FIPS), rather than being pulled toward a single global best result, termed as $gbest$, the particle's velocity is influenced by a weighted average of the personal best positions of all its neighbors. The personal best position of a neighbor is termed "exemplar" and is the aggregate information provided by the neighborhood space [15]. Comprehensive learning PSO (CLPSO) diversifies exemplar selection by allowing different dimensions of a particle to learn from different exemplars [16]. Diversity can be promoted via explicit dispersion or repulsion: ARPSO introduces a diversity-guided attractive–repulsive schedule to counter diversity collapse [17], and Monson and Seppi proposed adaptive diversity operators that trigger perturbations when the swarm becomes overly clustered [18]. A related form of disruption is leader replacement, where the best leader is periodically challenged or aged-out to escape local minima, as in PSO with an aging leader and challengers (ALC-PSO) [19].

C. Stagnation Detection and Adaptive Control

Deciding *when* to disrupt is often addressed via stagnation detectors. Clerc analyzed stagnation dynamics and emphasized that nonzero motion can coexist with negligible net displacement between the positions of the particles in the search space and therefore poor progress [11]. Worasucheep *et al.* proposed a PSO with stagnation detection and dispersion that triggers diversification actions when progress stalls [20]. Lihu and Holban further studied "disagreements on stagnation," using disagreement patterns among particles as a signal to initiate corrective actions [21]. Fitness-based stagnation detection can also be formalized; Ahmed *et al.* proposed an efficient criterion for identifying stagnation and triggering diversification while avoiding unnecessary disruption [22].

D. Topology Adaptation, Multi-swarms, and Annealing

Topology design is a long-standing approach to balance exploration and exploitation. Dynamic multi-swarm PSO (DMS-PSO) partitions the population into sub-swarms and periodically regroups them, refreshing exploration without a global restart [23]. A label, which is a discrete identifier or value, is assigned to each particle to denote it being a member of a specific sub-swarm. Other approaches adapt connectivity more continuously, for example via adaptive time-varying topology connectivity [24] or small-world rewiring strategies [25]. Recent multi-swarm variants also adapt the number of sub-swarms during the run to balance parallel exploration with later-stage exploitation [5].

E. Stability Analysis Under General Informer Models

Recent theory has formalized PSO stability for general *informer* formulations, where an *informer* is any particle (or neighborhood) whose best-so-far information influences another particle's update – i.e., it defines "who informs whom." An N-informer model particle aggregates guidance from N such informers (e.g., a neighborhood of size N). Cleghorn and Stapelberg derived order-1 and order-2 stability conditions for these models under broad coefficient distributions, clarifying how informer aggregation and parameter choices affect boundedness and convergence [26]. Related analyses exist for multi-guide and multi-swarm PSO algorithms [27]. These results motivate treating the informer graph as a first-class design variable. SD-PSO acts at this information-flow layer (corruption and migration) while keeping PSO control parameters fixed, making it complementary to parameter-tuning or stability-guided PSO methods rather than a replacement.

III. Adaptive Social-Dissent Particle Swarm Optimizer

This section discusses the structure of the social-dissent particle swarm optimizer (SD-PSO), detailing the operations constituting SD-PSO. Section III.A outlines the two dissent operators, i.e., the leader corruption operator, which makes a particle follow an alternative leader, and the sub-swarm membership migration operator, which temporarily changes the label of the particle to that of a different sub-swarm. Then the multi-swarm topology and annealing mechanism is discussed, which illustrates how the swarm breaks into sub-swarms and how it recombines into larger swarms. Section III.D outlines the method of using PLR to increase or decrease the probability of the dissent operators being applied. Section III.E presents the entire algorithm.

A. Dissent Operators

SD-PSO modifies the selection of the social leader $\boldsymbol{n}_i$ for a proportion of particles selected every iteration via two operators, as discussed below.

Leader corruption (I): Particles are grouped into $K(t)$ sub-swarms at iteration t. The operator forces a particle to follow a leader from a different sub-swarm for a single iteration. By effective probability $P_i^{I,\text{eff}}(t)$, particle i ignores its initially assigned group leader and instead follows the group best from a randomly sampled sub-swarm, denoted by $\boldsymbol{n}_j$.

Membership migration (M): The operator temporarily assigns a particle to a different sub-swarm for a single iteration. By effective probability $P_{(i)}^{M,\text{eff}}(t)$ (and only when $K(t) > 1$), particle i is temporarily treated as a member of a randomly chosen group $g_i' \sim \text{Unif}\{0, \ldots, K(t) - 1\}$. Its social guide is then taken as that group leader, i.e., $\mathbf{n}_i \leftarrow \mathbf{n}_{g_i'}$. This deliberately reroutes social identity and injects controlled disagreement without changing the core velocity update. When $K(t) = 1$, this operator is disabled by construction.

P^I and P^M are treated as *base rates* (static control parameters). The *effective* probabilities, $P_i^{I,\text{eff}}(t)$ and $P_{(i)}^{M,\text{eff}}(t)$, are computed by multiplying P_I and P_M by a factor $m_i(t)$. Section III.D defines $m_i(t)$ as the PLR-based multiplier that varies over time and across particles. Operators are evaluated sequentially; if both activate in the same iteration, membership migration overwrites leader corruption (M takes precedence).

B. Multi-swarm topology and annealing

SD-PSO's topology annealing uses a simple deterministic schedule, linearly reducing the number of active sub-swarms from $K_{\max}$ to 1. Particles are first partitioned into $K(t)$ sub-swarms. At initialization, each particle is randomly assigned to a sub-swarm. As $K(t)$ decreases, sub-swarms are *merged* deterministically by mapping the current sub-swarm label of the particles in the sub-swarm g_i to $g_i \bmod K(t)$. This yields a reproducible transition from multiple independent sub-swarms to aid in exploration to a single unified swarm which would aid in exploitation without imposing any index-based structure. $K(t)$ is annealed linearly over the evaluation budget from an initial $K(0) = K_0$ down to $K(t_{max}) = 1$; when $K(t) = 1$, the algorithm reduces to standard global-topology PSO. Because membership migration requires an alternate group, it is disabled by construction when $K(t) = 1$.

C. The Scout Operator

To reduce premature convergence and social stagnation, the scout operator (S) acts as a radical diversity mechanism. Instead of applying small perturbations to existing trajectories, it reassigns dissatisfied particles to new regions of the search space. A particle is considered for scouting based on its dissent probability $P_{dissent,i}(t)$, which is obtained by modulating the base scout probability P^S with the PLR multiplier $m_i(t)$.

A scout event is triggered for particle i at iteration t if $r < P_{dissent,i}(t), \quad r \sim \mathcal{U}[0, 1]$. When this occurs, the particle ignores both cognitive and social attraction terms and is reinitialized. Its position is updated for $j \in \{1, \ldots, d\}$ as

$$x_{ij}(t+1) = \begin{cases} \mathcal{U}(x_{\min,j}, x_{\max,j}) & \text{if } r < P_{dissent,i}(t) \\ x_{ij}(t) + v_{ij}(t+1) & \text{otherwise.} \end{cases} \tag{2}$$

Rather than setting the velocity to zero, the scout operator resamples it to remove residual momentum while preserving exploratory motion. For a particle entering scout mode, the scout-specific parameters are drawn as

$$w_i^{\text{scout}} \sim U(w_{\min}^{\text{scout}}, w_{\max}^{\text{scout}}), \qquad c_i^{\text{scout}} \sim U(c_{\min}^{\text{scout}}, c_{\max}^{\text{scout}}), \tag{3}$$

where $U(a, b)$ denotes the continuous uniform distribution on $[a, b]$.

The velocity is then resampled as

$$v_{ij}(t+1) \sim U(-v_{\max,j},\, v_{\max,j}) \qquad \text{if } r < P_{dissent,i}(t), \tag{4}$$

where $v_{\max,j} > 0$ is the velocity bound for dimension j.

Thus, the scout operator performs a stochastic long-range relocation in both position and velocity space. This discards the local search history of stagnant particles and restores exploratory capacity, improving the swarm's ability to escape local optima and sample previously unexplored basins.

D. Adaptive Noise via Path Length Ratio

Static dissent can help exploration, but can also harm convergence on some landscapes. Therefore, dissent is made *reactive* using a per-particle PLR, a "straightness" measure over a sliding window W (default $W = 8$). The per-particle PLR is calculated as

$$\text{PLR}_i(t) = \frac{\sum_{k=t-W}^{t-1} \|x_i(k+1) - x_i(k)\|}{\|x_i(t) - x_i(t-W)\| + \epsilon}. \tag{5}$$

Here, $\epsilon > 0$ is an infinitesimal value to avoid division-by-zero errors. A value near 1 indicates near-straight motion; values much greater than 1 indicate PLR oscillatory zig-zagging, which is used as a lightweight proxy for low net progress.

A *fresh* multiplier is computed for each iteration from the clipped PLR values, which is then used to scale the probabilities of leader corruption (P^I) and membership migration (P^M). The multiplier $m_i(t)$ is calculated as follows, where $\alpha > 0$ is a scaling factor that controls the sensitivity of the multiplier to deviations of $PLR_i(t)$ from one:

$$m_i(t) = \text{clip}\big(\alpha(\text{PLR}_i(t) - 1),\, 0,\, m_{\max}\big), \tag{6}$$

where $\text{clip}(x, a, b) = \min(\max(x, a), b)$. Then the base probabilities are scaled and clipped within the range $[0, 1]$ as follows:

$$P_i^{I,\text{eff}}(t) = \text{clip}\big(P^I\,(1 + m_i(t)),\, 0,\, 1\big), \tag{7a}$$

$$P_{(i)}^{M,\text{eff}}(t) = \text{clip}\big(P^M\,(1 + m_i(t)),\, 0,\, 1\big), \tag{7b}$$

$$P_{(i)}^{S,\text{eff}}(t) = \text{clip}\big(P^S\,(1 + m_i(t)),\, 0,\, 1\big). \tag{7c}$$

This creates negative feedback: oscillatory particles receive increased dissent, encouraging breakout, while steadily moving particles retain low dissent and can converge.

E. Full algorithm

The complete SD-PSO algorithm is presented in Algorithm 1.

Algorithm 1 Adaptive SD-PSO (PLR-scaled dissent with Scouts and Annealing)

Require: Objective f, bounds $[\mathbf{x}_{\min}, \mathbf{x}_{\max}]$, swarm size n, iterations T
Require: PSO params (ω, c_1, c_2); base rates (P^I, P^M, P^S)
Require: PLR params $(W, \alpha, m_{\max})$; sub-swarm schedule $K(t)$
1: Initialize particles $\{\mathbf{x}_i, \mathbf{v}_i\}_{i=1}^N$ and personal bests $\mathbf{p}_i \leftarrow \mathbf{x}_i$
2: Initialize PLR history buffers
3: Initialize sub-swarm labels $g_i \sim \text{Unif}\{0, \dots, K(0)-1\}$ for all particles
4: **for** $t = 1$ to T **do**
5: Update the active sub-swarm count $K(t)$ according to the annealing schedule; if $K(t) < K(t-1)$, merge labels via $g_i \leftarrow g_i \bmod K(t)$; identify group leaders $\ell(g)$
6: **for** $i = 1$ to n **do**
7: Compute $\text{PLR}_i(t)$ using Eq. (5)
8: Compute $m_i(t)$ using Eq. (6)
9: Compute $P_{(i)}^{S,\text{eff}}$ using Eq. (7c)
10: **if** rand() $< P_{(i)}^{S,\text{eff}}$ **then**
11: $\mathbf{x}_i \leftarrow \mathcal{U}(\mathbf{x}_{\min}, \mathbf{x}_{\max})$
12: Compute w_i^{scout} and c_i^{scout} based on Eq. (3)
13: $\mathbf{v}_i \leftarrow \mathcal{U}(-\mathbf{v}_{\max}, \mathbf{v}_{\max})$
14: **else**
15: $\mathbf{n}_i \leftarrow \mathbf{p}_{\ell(g_i)}$
16: $P_i^{I,\text{eff}} \leftarrow \text{clip}(P^I(1 + m_i(t)), 0, 1)$
17: **if** rand() $< P_i^{I,\text{eff}}$ **then**
18: $\mathbf{n}_i \leftarrow \mathbf{p}_{j \sim \mathcal{U}(1,N)}$
19: **end if**
20: $P_{(i)}^{M,\text{eff}} \leftarrow \text{clip}(P^M(1 + m_i(t)), 0, 1)$
21: **if** $K(t) > 1$ and rand() $< P_{(i)}^{M,\text{eff}}$ **then**
22: $\mathbf{n}_i \leftarrow \mathbf{p}_{\ell(g')}$, where $g' \neq g_i$
23: **end if**
24: **end if**
25: $\mathbf{v}_i$ computed as in Eq. (1)
26: $\mathbf{x}_i(t) \leftarrow \mathbf{x}_i(t-1) + \mathbf{v}_i$
27: Evaluate the objective $f(\mathbf{x}_i)$; update $\mathbf{p}_i$ if an improvement is found
28: **end for**
29: **end for**

The adaptive SD-PSO initiates each iteration by partitioning the swarm into $K(t)$ independent sub-swarms to facilitate parallel exploration (line 3). For every particle, the algorithm computes the PLR over a sliding window (line 7) to quantify kinematic stability (i.e., the efficiency and directionality of the particle's trajectory through the search space). This metric drives the calculation of a dissatisfaction multiplier $m_i(t)$ (line 8), which dynamically scales the effective probabilities for the dissent operators (lines 9, 16, and 20). The logic first evaluates the scout condition (line 10); if triggered, the particle undergoes a stochastic position re-initialization and scout-velocity resampling, with scout control parameters sampled from their prescribed ranges, to be taken into account in the next iteration (lines 11–13), enabling a long-jump out of the current basin. Otherwise, the particle remains within the social structure and defaults to tracking its assigned group leader (line 15). This target may be overridden by a random global informer (the information corruption operator) (lines 17–19) or by a neighboring group leader (lines 21–23) based on the scaled dissent probabilities. The particle's state is then updated via the standard kinematic equations using the final resolved social exemplar (lines 25–26), followed by objective function evaluation and personal-best update (line 27).

IV. Experimental Setup

This section outlines the implementation details and experimental setup. Section IV.A describes the benchmark protocol, baselines, success definition, and control parameters. Section IV.B defines the common task set used for strict cross-algorithm comparability. Section IV.C defines the relative expected running time aggregate used in the stability maps.

Throughout the analysis, the notation `IX MX SX AX` is employed, where `I`, `M`, `S`, and `A` refer to informer corruption, membership migration, unstable scouts, and topological annealing, respectively. The variable `X` takes the value `0` or `1` depending on whether the method was enabled or disabled; for instance, `I0 M1 S0 A1` represents a variant utilizing only membership migration and topological annealing.

A. Experimental Protocol, Baselines, and Success Definition

The experiments use the COCO/BBOB noiseless test suite with COCO post-processing for expected runtime (ERT) and success statistics [6], [28]. All methods use equal evaluation budgets of $10^4 \cdot d$ function evaluations per run, with matched random seeds. A run is counted as successful if it reaches the hardest "solved" precision threshold $\Delta f \leq 10^{-8}$ within the evaluation budget.

The control parameter values used are listed in Table I. The core PSO parameters follow widely used settings for inertia-weight PSO and were kept fixed across all experiments to isolate the effect of the dissent operators. The dissent-related parameters were selected through preliminary pilot experiments and coarse grid exploration, with the aim of identifying stable values that performed consistently across the benchmark set rather than values tailored to individual functions or dimensions.

The study compares against the BBOB-2009 PSO by El-Abd and Kamel [8] and DE-PSO [9]. In addition, strong reference results, such as IPOP-ACTCMA-ES, and other PSO hybrids are included as reported in the COCO/BBOB archive [28]. As an internal code-validation control, a "vanilla" SD-PSO variant with dissent and annealing disabled is also executed; its purpose is implementation sanity checking rather than serving as a headline comparator.

All experiments follow the standardized COCO/BBOB black-box benchmarking protocol, including the definition of targets, success, and aggregation of expected running time across functions, dimensions, and instances [6], [7]. Relative ERT is analyzed on a log scale, and success rate is reported at

TABLE I
CONTROL PARAMETERS FOR SD-PSO

Category	Parameter	Value
Core PSO	Inertia weight (w)	0.72
	Acceleration coeff. (c_1, c_2)	1.49
	Velocity limit (v_{max})	2.0
	Population size (N)	40
Noise Rates (Base)	Informer Corruption (P^I)	0.05
	Membership Migration (P^M)	0.10
	Unstable Scouts (P^S)	0.10
Topology Annealing	Initial Sub-swarms (K_0)	6
	Final Sub-swarms (K_{final})	1
Scout Behavior	Inertia Range (w_{scout})	$[-0.2, 1.2]$
	Coeff. Range (c_{scout})	$[0.0, 4.0]$
PLR Mechanism	Window Size (W)	8
	Scaling factor (α)	0.05
	Multiplier Cap	0.75

the hardest target, enabling reproducible comparisons against the postprocessed archive baselines under the same target and relative-ERT conventions. Pairwise Wilcoxon p-values are reported later in Section V.C for selected comparisons.

B. Common task set

For strict cross-algorithm comparability, results are additionally reported on the *intersection* of tasks for which all included algorithms have recorded results, i.e., valid runtime/ERT entries are available in the COCO/BBOB post-processing. This *common task set* contains 840 tasks, comprising the 24 noiseless BBOB functions evaluated at five dimensions $d \in \{2, 3, 5, 10, 20\}$, with 7 runs per function – dimension pair ($24 \times 5 \times 7$). Restricting analysis to this shared subset ensures that convergence-rate comparisons are made on exactly the same function, dimension, and instance workloads across all methods.

C. Relative Expected Running Time Aggregate Used in Stability Maps

To visualize stability over the dissent control parameters (P^I, P^M), runtimes are summarized across tasks using a log-scale aggregation consistent with BBOB practice for heavy-tailed runtime distributions. Concretely, for a task set $\mathcal{T}$, the relative expected running time (rERT), defined as the ratio of an algorithm's expected running time to that of a fixed reference algorithm, is

$$\mathrm{rERT}_\tau = \frac{\mathrm{ERT}_\tau^{(\mathrm{alg})}}{\mathrm{ERT}_\tau^{(\mathrm{ref})}}. \tag{8}$$

In the present study, the reference algorithm is IPOP-ACTCMA-ES. The aggregate score used in the stability maps is then defined as

$$\mathrm{Score}_{\mathrm{rERT}} = \exp\left(\frac{1}{|\mathcal{T}|}\sum_{\tau\in\mathcal{T}} \log\left(\mathrm{rERT}_\tau\right)\right). \tag{9}$$

Smaller values are better. For example, a score of 1.8 corresponds to a geometric-mean relative ERT of 1.8. Figure 1 plots this score over a coarse grid search for two cases: annealing disabled and annealing enabled.

While aggregating performance metrics across varied dimensions and function topologies provides a broad measure of general algorithmic robustness, it can obscure dimension-specific behaviors. The geometric mean of relative ERT is deliberately employed here to mitigate vast scale disparities. Because optimization runtimes across BBOB tasks typically span several orders of magnitude, an arithmetic mean would be heavily skewed by the most deceptive or highest-dimensional tasks. The geometric aggregation ensures that relative efficiency gains on simpler tasks are evaluated proportionally alongside improvements on complex ones. However, the limits of interpreting a single cross-dimensional aggregate are acknowledged. To safeguard against over-generalization, the primary claims regarding the mitigation of social lock-in are not based solely on this aggregate score. Instead, the results are also disaggregated across dimensions in Section V.D.

V. RESULTS

This section presents the results of the ablation study based on the four noise-introducing methods outlined, alongside an analysis of the data generated over the execution of the variants on the `COCO/BBOB` benchmarking suite.

Section V.A details the stability study performed over P^I and P^M and the optimal parameter set derived from this analysis. The subsequent section examines the general performance relative to the algorithms considered for benchmarking, while also providing data for the best-performing variants. The validity of the introduction of noise changing model dynamics is analyzed using pairwise Wilcoxon p-values in section V.C. Next, the results in higher dimensions are presented, justifying their separate examination before discussing the specific contexts where dissent proves beneficial.

A. Stability maps

Figure 1 illustrates how topology annealing expands the stable control parameter region within the (P^I, P^M) space. This specific region represents the area where the algorithm achieves its peak performance.

For configurations without annealing, the $(P^I = 0, P^M = 0)$ setting results in a high geometric mean for the relative ERT aggregate, with a score of approximately 1.844.

In contrast, the introduction of annealing reduces the algorithm's sensitivity to control parameter selection. This process widens the stable regions, as demonstrated by the increased number of parameter sets achieving the best performance score of 1.582. Ultimately, these results suggest that scheduled multi-swarm consolidation serves as an effective complement to swarm dissent.

1) Control Parameter Sensitivity Analysis: The algorithm's robustness is quantified to the dissent base rates P^I (leader corruption) and P^M (membership migration) by analyzing the volatility of the relative ERT scores across the grid search in Figure 1. Sensitivity is defined as the performance penalty incurred by deviating from optimal control parameters to the baseline state ($P^I = P^M = 0$). The results with and without using annealing are outlined as follows,

- **High Sensitivity (Annealing OFF):** Without topology annealing, the algorithm exhibits a "cliff-edge" sensitivity

profile as the performance degrades by **15.4%** when moving from optimal configuration (score: 1.598) to baseline (score: 1.844). The large difference between the highest and lowest values of $rERT$ (Δ_{ERT}) which is ≈ 0.246 indicates that precise tuning is critical for success.

- **Robustness (Annealing ON):** The performance penalty for reverting to the baseline drops to just **4.2%** (score: 1.582 vs 1.649). The significantly narrower range ($\Delta_{ERT} \approx 0.079$) confirms that annealing renders the algorithm largely insensitive to specific noise rates, ensuring reliable convergence even with sub-optimal settings.

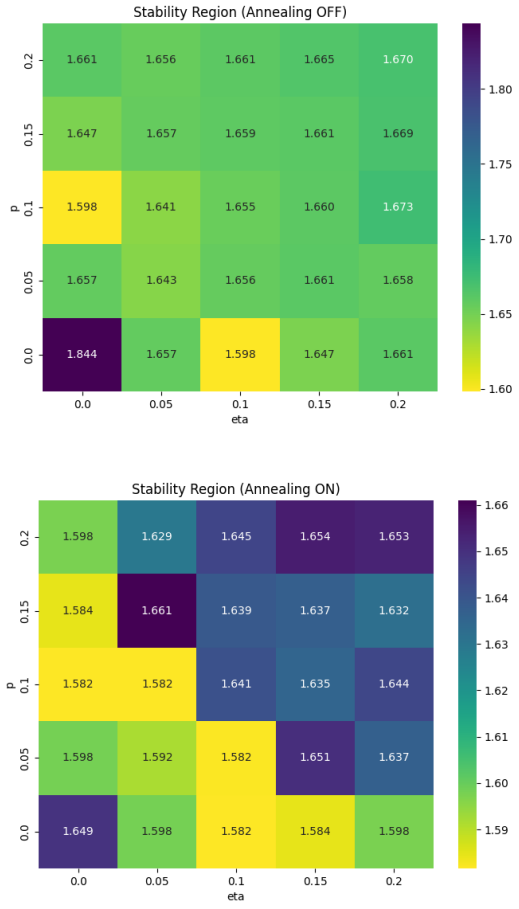

Fig. 1. Grid search stability maps using a geometric-mean of relative ERT score (lower is better). Top: annealing OFF. Bottom: annealing ON.

B. Aggregate performance

Table II reports the geometric mean of ERT scores and single-target success rates on the common task set of 840 tasks. The results demonstrate that the SD-PSO variants as displayed in table II consistently outperform the standard PSO baseline (BBOB-2009) in terms of computational efficiency and robustness.

The top-performing configuration, SD-PSO `(I0 M1 S1 A1)` (combining cross-group consultation, scouts, and annealing), achieves a geometric mean of ERT Score of **36.64** and a success rate of **0.31**. This represents a substantial improvement over the PSO BBOB-2009 baseline, which records a score of 43.93 and a success rate of 0.24. Furthermore, the stable configuration without scouts (`I0 M1 S0 A1`) also surpasses the baseline with a score of 40.63 and a success rate of 0.28, confirming that the core mechanisms of consultation and annealing provide robust benefits independent of reinitialization strategies.

In comparative terms, the top SD-PSO variants occupy the performance tier immediately following the state-of-the-art IPOP-ACTCMA-ES (Score: 2.53), while successfully outranking in convergence rates other hybrid benchmarks such as EvoSpace-PSO-GA (Score: 43.37) and DE-PSO (Score: 61.68) when all of them reach optima. Table III confirms these trends in log-space, where the proposed method reduces the mean log score from 1.64 (for the baseline) to 1.56, indicating a consistent shift toward lower runtimes across the task distribution.

TABLE II
COMPARISON OF GEOMETRIC MEAN OF ERT SCORES ON THE COMMON TASK SET (840 TASKS). SMALLER VALUES ARE BETTER.

Algorithm	Geo-Mean Score	Success Rate
IPOP-ACTCMA-ES (Baseline)	**2.53**	**0.82**
SD-PSO (I0 M1 S1 A1)	**36.64**	**0.31**
SD-PSO (I1 M1 S0 A1)	38.92	0.29
SD-PSO (I0 M0 S1 A1)	39.66	0.28
SD-PSO (I1 M0 S1 A1)	40.12	0.28
SD-PSO (I0 M1 S0 A1)	40.63	0.28
SD-PSO (I0 M0 S0 A1)	42.73	0.27
EvoSpace-PSO-GA	43.37	0.34
PSO (BBOB-2009)	43.93	0.24
DE-PSO	61.68	0.14

TABLE III
LOG-SCALE PERFORMANCE SUMMARY (MEAN LOG RELATIVE ERT). SMALLER VALUES ARE BETTER.

Algorithm	Mean Log Score	Success Rate
IPOP-ACTCMA-ES	0.40	0.82
SD-PSO (I0 M1 S1 A1)	**1.56**	**0.31**
SD-PSO (I1 M1 S0 A1)	1.59	0.29
SD-PSO (I0 M0 S1 A1)	1.60	0.28
SD-PSO (I1 M0 S1 A1)	1.60	0.28
SD-PSO (I0 M1 S0 A1)	1.61	0.28
EvoSpace-PSO-GA	1.64	0.34
PSO (BBOB-2009)	1.64	0.24
DE-PSO	1.79	0.14

C. Pairwise Wilcoxon P-Values

To compare the reported variants with the standard PSO baseline, a pairwise Wilcoxon signed-rank test was applied to the selected configurations. Figure 2 reports the resulting p-values for the pairwise comparisons. Lower p-values indicate stronger incompatibility with the corresponding null hypothesis for the tested pair, but they do not by themselves measure effect size or practical importance.

Across the comparisons against the baseline, the reported p-values are small for the selected SD-PSO variants. For example, the baseline comparison with `I0 M1 S1 A1` yields $p \approx 1.3510^{-6}$, while the comparison with `I0 M1 S0 A1` yields $p \approx 1.9410^{-4}$. These results indicate that the observed performance differences are unlikely under the corresponding null hypotheses for those paired comparisons.

The internal comparisons among SD-PSO variants also return small p-values for some pairs. These results suggest that the choice of dissent operators affects the empirical behavior of the algorithm. However, the p-values should be interpreted together with the reported runtime and success-rate summaries in Tables II–IV.

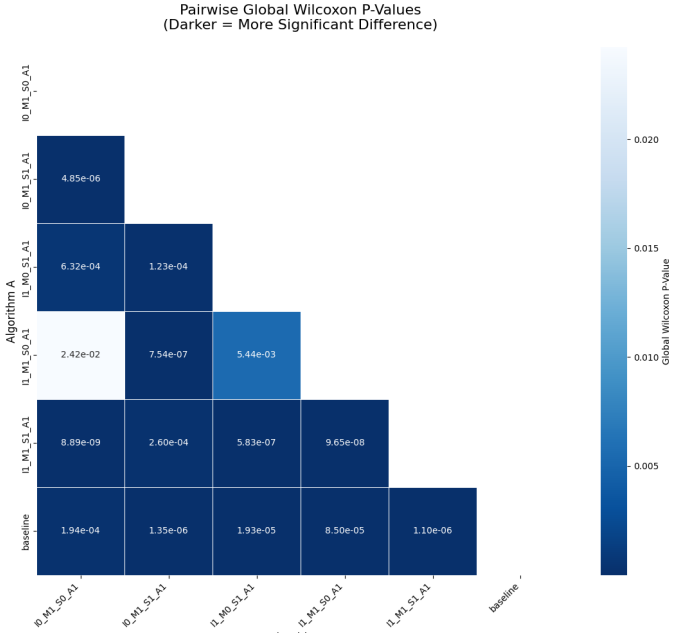

Fig. 2. Pairwise Wilcoxon p-values for the selected algorithm comparisons.

D. Performance in Higher Dimensions

The impact of social dissent becomes increasingly pronounced as the dimensionality of the search space increases. While the aggregate results across all dimensions (with $d \in \{2, \ldots, 20\}$) showed a moderate improvement over the standard baselines, a focused analysis on the higher-dimensional subset ($d = 10$ and $d = 20$) reveals a substantial widening of the performance gap between the proposed SD-PSO variants and established PSO benchmarks.

As detailed in Table IV, the standard PSO [8] achieves a success rate of only 4.2% on this high-dimensional task set, indicating a near-total collapse of exploration capabilities. Similarly, the hybrid DE-PSO benchmark falters with a success rate of just 6.3%. In stark contrast, the top-performing SD-PSO configuration (`I0 M1 S0 A1`) maintains a success rate of 16.7%, a relative improvement of approximately four times that of the standard PSO baseline.

This divergence suggests that while standard social learning mechanisms are sufficient for low-dimensional navigation, they rapidly succumb to social lock-in as the volume of the search space expands. The dissent operators effectively counteract this stagnation by enforcing information exchange between disjoint sub-populations, preserving a higher success rate where traditional methods fail. The robustness of these results at $d = 20$ provides strong motivation for extending the SD-PSO framework to even higher-dimensional problems ($d = 20$).

TABLE IV
PERFORMANCE ON HIGH-DIMENSIONAL TASKS ($d \in \{10, 20\}$). THE TABLE COMPARES TOP SD-PSO VARIANTS AGAINST EXTERNAL BENCHMARKS. LOWER LOG SCORE IS BETTER; HIGHER SUCCESS RATE IS BETTER.

Algorithm Configuration	**Weighted Log Score**	**Success Rate**
State-of-the-Art Reference		
IPOP-ACTCMA-ES	0.50	0.75
Proposed SD-PSO Variants		
`I0 M1 S0 A1` (Consult + Anneal)	**1.82**	**0.17**
`I1 M0 S0 A1` (Corrupt + Anneal)	1.84	0.15
`I1 M0 S0 A0` (Corrupt only)	1.84	0.15
`I0 M1 S0 A0` (Consult only)	1.84	0.15
`I1 M1 S0 A1` (All + Anneal)	1.85	0.15
External Baselines		
DE-PSO (Garcia-Nieto)	1.93	0.06
PSO (El-Abd & Kamel 2009)	1.96	0.04

E. Where dissent helps

Across the stability maps and aggregate summaries, the strongest gains were observed on landscapes where early leader lock-in is common. Topology annealing expands the viable operating region by reducing sensitivity to (P^I, P^M) choice, and PLR-adaptation prevents dissent from being uniformly active when particles are already making coherent progress. The combined (`I1 M1 S0 A1`) variant achieves higher success in Table III but has slightly worse mean rank; (`I0 M1 S0 A1`) is highlighted as a simpler configuration which increases success rate.

VI. CONCLUSION

This study addressed the vulnerability of particle swarm optimization (PSO) to groupthink and social lock-in by introducing the adaptive social-dissent particle swarm optimization (SD-PSO). The proposed algorithm integrates three distinct dissent mechanisms, leader corruption, membership migration, and the scout operator to inject adaptive stochastic noise into the social learning process. By utilizing a per-particle path length ratio (PLR) to quantify kinematic stability, the algorithm dynamically scales dissent probabilities, transforming stagnation detection into a continuous control signal rather than a binary trigger. Furthermore, the implementation of topology annealing ensures a deterministic transition from multi-swarm exploration to unified exploitation, effectively balancing diversity with convergence.

Extensive benchmarking on the COCO/BBOB noiseless test suite demonstrates that SD-PSO achieved superior single-target success rates and geometric mean of relative expected runtime (ERT) scores compared to the BBOB-2009 PSO baseline and

hybrid variants, such as DE-PSO. This performance advantage is particularly pronounced in higher-dimensional tasks, where the dissent operators effectively counteract diversity collapse. Stability map analysis confirms that topology annealing acts as a sensitivity buffer, broadening the robust operating region for control parameter selection. Finally, pairwise Wilcoxon signed-rank tests provide strong evidence of these changes altering the algorithm. When the evidence is combined by better performance on benchmarks, we make the case that PLR-modulated social dissent enables the swarm to escape local optima without permanently destabilizing the search dynamics.

Future Work: Future research will focus on a detailed analysis of the search dynamics and trajectory characteristics of SD-PSO. Additional investigations will assess its scalability and performance on large-scale, high-dimensional optimization problems. Moreover, an extensive sensitivity analysis of the complete set of control parameters will be conducted to better understand their impact on algorithmic behavior and performance.

References

[1] J. Kennedy, *Particle Swarm Optimization*. Boston, MA: Springer US, 2010, pp. 760–766. [Online]. Available: https://doi.org/10.1007/978-0-387-30164-8_630

[2] F. van den Bergh and A. Engelbrecht, "A study of particle swarm optimization particle trajectories," *Information Sciences*, vol. 176, no. 8, pp. 937–971, 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025505000630

[3] H. Wang, H. Sun, C. Li, S. Rahnamayan, and J. shyang Pan, "Diversity enhanced particle swarm optimization with neighborhood search," *Information Sciences*, vol. 223, pp. 119–135, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025512006779

[4] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *Trans. Evol. Comp*, vol. 6, no. 1, p. 58–73, Feb. 2002. [Online]. Available: https://doi.org/10.1109/4235.985692

[5] X. Xia, L. Gui, and Z.-H. Zhan, "A multi-swarm particle swarm optimization algorithm based on dynamical topology and purposeful detecting," *Applied Soft Computing*, vol. 67, pp. 126–140, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494618301017

[6] N. Hansen, S. Finck, R. Ros, and A. Auger, "Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions," INRIA, Research Report RR-6829, 2009. [Online]. Available: https://inria.hal.science/inria-00362633

[7] N. Hansen, A. Auger, R. Ros, O. Mersmann, T. Tušar, and D. Brockhoff, "Coco: a platform for comparing continuous optimizers in a black-box setting," *Optimization Methods and Software*, vol. 36, no. 1, pp. 114–144, 2021. [Online]. Available: https://doi.org/10.1080/10556788.2020.1808977

[8] M. El-Abd and M. S. Kamel, "Black-box optimization benchmarking for noiseless function testbed using particle swarm optimization," in *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, ser. GECCO '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 2269–2274. [Online]. Available: https://doi.org/10.1145/1570256.1570316

[9] J. García-Nieto, E. Alba, and J. Apolloni, "Particle swarm hybridized with differential evolution: black box optimization benchmarking for noisy functions," in *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, ser. GECCO '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 2343–2350. [Online]. Available: https://doi.org/10.1145/1570256.1570327

[10] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, vol. 2, 2002, pp. 1671–1676 vol.2.

[11] M. Clerc, "Stagnation Analysis in Particle Swarm Optimisation or What Happens When Nothing Happens," Dec. 2006, 17 pages. [Online]. Available: https://hal.science/hal-00122031

[12] T. Blackwell and J. Branke, "Multiswarms, exclusion, and anti-convergence in dynamic environments," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 459–472, 2006.

[13] D. Freitas, L. G. Lopes, and F. Morgado-Dias, "Particle swarm optimisation: A historical review up to the current developments," *Entropy*, vol. 22, no. 3, 2020. [Online]. Available: https://www.mdpi.com/1099-4300/22/3/362

[14] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36221)*. IEEE, 1998, pp. 69–73.

[15] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better," *Trans. Evol. Comp*, vol. 8, no. 3, p. 204–210, Jun. 2004. [Online]. Available: https://doi.org/10.1109/TEVC.2004.826074

[16] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *Trans. Evol. Comp*, vol. 10, no. 3, p. 281–295, Sep. 2006. [Online]. Available: https://doi.org/10.1109/TEVC.2005.857610

[17] J. Vesterstrøm and J. Riget, "A diversity-guided particle swarm optimizer - the arpso," *EVALife Technical Report*, no. 2002-02, 2002.

[18] C. K. Monson and K. D. Seppi, "Adaptive diversity in pso," in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 59–66. [Online]. Available: https://doi.org/10.1145/1143997.1144006

[19] W.-N. Chen, J. Zhang, Y. Lin, N. Chen, Z.-H. Zhan, H. S.-H. Chung, Y. Li, and Y.-H. Shi, "Particle swarm optimization with an aging leader and challengers," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 2, pp. 241–258, 2013.

[20] C. Worasucheep, "A particle swarm optimization with stagnation detection and dispersion," in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, 2008, pp. 424–429.

[21] A. Lihu, Ş. Holban, and O.-A. lihu, *Particle Swarm Optimization with Disagreements on Stagnation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 171–190. [Online]. Available: https://doi.org/10.1007/978-3-642-53878-0_9

[22] H. R. Ahmed, "An efficient fitness-based stagnation detection method for particle swarm optimization," in *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO Comp '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 1029–1032. [Online]. Available: https://doi.org/10.1145/2598394.2605669

[23] J. Liang and P. Suganthan, "Dynamic multi-swarm particle swarm optimizer," in *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005.*, 2005, pp. 124–129.

[24] W. H. Lim and N. A. Mat Isa, "Particle swarm optimization with adaptive time-varying topology connectivity," *Applied Soft Computing*, vol. 24, pp. 623–642, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494614003780

[25] Y.-j. Gong and J. Zhang, "Small-world particle swarm optimization with topology adaptation," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 25–32. [Online]. Available: https://doi.org/10.1145/2463372.2463381

[26] C. W. Cleghorn and B. Stapelberg, "Particle swarm optimization: Stability analysis using n-informers under arbitrary coefficient distributions," *Swarm and Evolutionary Computation*, vol. 71, p. 101060, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210650222000323

[27] C. Scheepers, A. P. Engelbrecht, and C. W. Cleghorn, "Multi-guide particle swarm optimization for multi-objective optimization: empirical and stability analysis," *Swarm Intelligence*, vol. 13, no. 3, pp. 245–276, dec 2019. [Online]. Available: https://doi.org/10.1007/s11721-019-00171-0

[28] COCO Platform, "Algorithm data sets for the bbob test suite," https://coco-platform.org/testsuites/bbob/data-archive.html, 2024, accessed: 2026-04-11.