HOKKAIDO UNIVERSITY

Title	Dynamic fitness-distance balance competitive swarm optimizer : performance investigation, engineering Simulation, and application in superconductor critical temperature prediction
Author(s)	Cao, Yang; Du, Xingbang; Yu, Jun et al.
Citation	Cluster Computing: The Journal of Networks Software Tools and Applications, 28(11), 745 https://doi.org/10.1007/s10586-025-05366-1
Issue Date	2025-09-11
Doc URL	https://hdl.handle.net/2115/98114
Rights	This version of the article has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: https://doi.org/10.1007/s10586-025-05366-1
Type	journal article
File Information	DFDBCSO.pdf



Hokkaido University Collection of Scholarly and Academic Papers : HUSCAP

Dynamic Fitness-Distance Balance Competitive Swarm Optimizer: Performance Investigation, Engineering Simulation, and Application in Superconductor Critical Temperature Prediction

Yang Cao[a], Xingbang Du[a], Jun Yu[b], Rui Zhong[c,*] and Masaharu Munetomo[c]

[a]Graduate School of Information Science and Technology, Hokkaido University, Sapporo, Japan
[b]Institute of Science and Technology, Niigata University, Niigata, Japan
[c]Information Initiative Center, Hokkaido University, Sapporo, Japan

ARTICLE INFO

Keywords:
Competitive Swarm Optimizer
Dynamic Fitness-Distance Balance
CEC Benchmarks
Superconductors
Critical temperature prediction

ABSTRACT

This paper introduces the Dynamic Fitness-Distance Balance (DFDB) strategy to enhance the Competitive Swarm Optimizer (CSO), resulting in the proposed DFDBCSO algorithm. DFDBCSO improves exploration efficiency and solution accuracy by dynamically balancing individual fitness and population diversity throughout the optimization process. In the early exploration phase, DFDB emphasizes solution diversity by encouraging competition based on the distance from the best solution (P_{best}), fostering a balance between active and less active individuals. In the later stages, the focus shifts toward fitness-based competition, accelerating convergence by prioritizing interactions between high- and low-performing individuals. We conducted comparative experiments on the CEC2017, CEC2020, and CEC2022 benchmark functions against other algorithms. On the CEC2017 benchmark, DFDBCSO achieved the highest average ranking in the 30-dimensional setting and ranked second in both the 50-dimensional and 100-dimensional settings. For CEC2020, DFDBCSO secured the top average ranking across the 10-dimensional, 20-dimensional, and 100-dimensional settings. Similarly, on the CEC2022 benchmark, DFDBCSO achieved the highest average ranking in both the 10-dimensional and 20-dimensional settings. Additionally, we evaluated the performance of DFDBCSO on five engineering design problems. DFDBCSO ranked first on the CBDP, REBDP, and WBDP problems, while it ranked second and third on the SRDP and TCSDP problems, respectively. Furthermore, DFDBCSO is integrated into an ensemble regression model to predict the critical temperature of superconductors using atomic and physicochemical properties. The proposed DFDBCSO-Ensemble model achieves the highest prediction accuracy, outperforming the second-ranked ExtraTrees model by reducing the Mean Squared Error (MSE) by 1.82, decreasing the Mean Absolute Error (MAE) by 0.04, and improving the R-squared score (R^2) by 0.01.

1. Introduction

Optimization algorithms are essential for addressing complex challenges in diverse domains such as engineering design, machine learning, data mining, and scientific research [1, 2]. As the scale and complexity of problems continue to grow, finding globally optimal or near-optimal solutions within limited computational resources has become a key challenge for researchers and engineers [3–5]. Heuristic algorithms are highly efficient, adaptable, and easy to implement. They can quickly find acceptable approximate solutions for complex, high-dimensional, and NP-hard problems. Recent heuristic algorithms include the Potter Optimization Algorithm (POA) [6], Carpet Weaving Optimization (CWO) [7], Fossa Optimization Algorithm (FOA) [8], Addax Optimization Algorithm (AOA) [9], Dollmaker Optimization Algorithm (DOA) [10], Sales Training Based Optimization (STBO) [11], Orangutan Optimization Algorithm (OOA) [12], Spider-Tailed Horned Viper Optimization (STHVO) [13], among others.

Optimization algorithms play a crucial role in engineering design, load forecasting, and energy system optimization. For example, In the multidisciplinary design optimization of a semi-submersible platform's dynamic positioning system, the Alpine Skiing Optimization (ASO) algorithm is used to solve the optimization problem by coordinating

*Corresponding author
yang.cao.y4@elms.hokudai.ac.jp (Y. Cao); xingbang.du.d1@elms.hokudai.ac.jp (X. Du); yujun@ie.niigata-u.ac.jp (J. Yu); zhongrui@iic.hokudai.ac.jp (R. Zhong); munetomo@iic.hokudai.ac.jp (M. Munetomo)
ORCID(s): 0009-0009-8394-1474 (Y. Cao); 0009-0007-3117-6083 (X. Du); 0000-0001-5029-0294 (J. Yu); 0000-0003-4605-5579 (R. Zhong); 0000-0002-5750-9217 (M. Munetomo)

Table 1
Comparison of metaheuristic algorithms and their limitations.

Algorithm	Main Contribution	Limitations
Genetic Algorithm (GA) [32]	Crossover and mutation for global search	Slow convergence, high parameter sensitivity
Particle Swarm Optimization (PSO) [33]	Swarm-based information sharing	Premature convergence in multimodal problems
Differential Evolution (DE) [34]	Adaptive mutation strategies	Sensitive to parameter tuning
Competitive Swarm Optimizer (CSO) [20]	Pairwise competition mechanism	Rigid update rules, lack of adaptability
DFDBCSO (Proposed)	Dynamic adaptation of competition criteria	Increased computational overhead due to dynamic adaptation

structural and dynamic performance parameters, improving system stability, and reducing operational costs [14]. In short-term power load forecasting, the Sparrow Search Optimization (SSO) algorithm is employed to optimize hyperparameters in machine learning models, enhancing prediction accuracy and demonstrating the effectiveness of ensemble learning methods in load forecasting [15]. In fuel cell parameter identification, the Attack-Defense Strategy Assisted Osprey Optimization Algorithm (ADSOOA) improves convergence performance and avoids local optima, enabling efficient parameter optimization in PEMFC systems [16]. These studies highlight the broad application potential of optimization algorithms in engineering, energy, and intelligent systems.

Among numerous optimization algorithms, swarm intelligence algorithms have emerged as a research hotspot due to their bioinspired nature and adaptability to nonlinear high-dimensional problems [17–19]. Competitive Swarm Optimizer (CSO) [20] is one of the most prominent swarm intelligence algorithms, widely applied in engineering optimization, image processing, and energy management due to its simple structure, high parallelism, and powerful search capabilities [21–23].

Despite its advantages, CSO still faces several challenges when applied to complex optimization problems. First, it suffers from premature convergence, often getting trapped in local optima, especially in multimodal and complex search spaces [24, 25]. This occurs because the algorithm lacks mechanisms to maintain sufficient population diversity in the later stages of optimization, reducing its ability to escape local minima and find better solutions [26, 27]. Second, CSO lacks dynamic adaptation mechanisms in its competition strategy. In traditional CSO, weaker individuals always move toward stronger ones based on a fixed competition mechanism [28]. However, this rigid approach fails to adapt to different optimization stages, leading to an imbalance between exploration and exploitation [29–31]. The inability to flexibly adjust competition criteria based on population dynamics limits CSO's effectiveness in solving diverse and complex problems.

Existing studies have attempted to address these issues by introducing modifications to CSO, such as adaptive parameter tuning or hybridization with other metaheuristic strategies. However, most improvements focus on either enhancing exploration or exploitation, rather than dynamically balancing both throughout the optimization process. A more effective solution should regulate search behavior dynamically based on problem characteristics and optimization progress.

To overcome these limitations, we propose the Dynamic Fitness-Distance Balance-based Competitive Swarm Optimizer (DFDBCSO), which introduces a novel Dynamic Fitness-Distance Balance (DFDB) strategy to enhance the competition mechanism of CSO. DFDB dynamically adjusts competition criteria by considering both individual fitness and spatial distance from the best solution (P_{best}), ensuring a smooth transition from exploration to exploitation. During the early phase of optimization, DFDB prioritizes diversity by encouraging competition based on distance, allowing individuals far from P_{best} to explore new regions. As optimization progresses, the focus gradually shifts to fitness-based competition, refining the search process and accelerating convergence. This two-phase competition framework enables DFDBCSO to maintain population diversity while improving solution accuracy, effectively addressing the weaknesses of traditional CSO.

To highlight the advantages of DFDBCSO, Table 1 presents a comparison of various metaheuristic algorithms, their primary contributions, and their limitations. DFDBCSO is designed to overcome the shortcomings of previous methods by introducing a dynamic adaptation mechanism, significantly improving its performance in complex problem landscapes.

The effectiveness of DFDBCSO is validated through extensive experiments on CEC2017, CEC2020, and CEC2022 benchmark functions, where it consistently outperforms state-of-the-art algorithms. Additionally, DFDBCSO is evaluated on five real-world engineering design problems, demonstrating its robustness across diverse optimization tasks. Furthermore, we apply DFDBCSO to a machine learning regression task, constructing an ensemble model for

predicting the critical temperature of superconductors using atomic and physicochemical properties. The DFDBCSO-Ensemble model achieves state-of-the-art performance, significantly reducing Mean Squared Error (MSE) and improving predictive accuracy compared to traditional regression models.

The key contributions of this research are summarized as follows:

1. **Development of an adaptive competition mechanism for CSO:** We introduce the DFDB strategy, which dynamically balances exploration and exploitation by adjusting competition criteria based on fitness and spatial distance.
2. **Extensive benchmarking and real-world validation:** DFDBCSO is systematically evaluated on CEC benchmark functions and engineering optimization problems, confirming its superior performance over existing methods.
3. **Application to machine learning regression tasks:** DFDBCSO is integrated into an ensemble regression model, significantly improving prediction accuracy for superconducting critical temperatures.

The remainder of this paper is organized as follows: Section 2 reviews related work, while Section 3 details the proposed DFDBCSO algorithm. Numerical experiments using CEC benchmarks are presented in Section 4, followed by engineering problem experiments in Section 5. Section 6 examines DFDBCSO's application to predicting superconductor critical temperatures. We discussed the results of DFDBCSO in different experiments in Section 7. Finally, Section 8 concludes the paper and suggests future research directions.

2. Related works

2.1. Fitness-Distance Balance(FDB)

Fitness-Distance Balance (FDB) is a critical metric for evaluating optimization algorithms by integrating solution fitness with the distance to the optimal point [35]. It evaluates the population's capacity to navigate the trade-off between exploration and exploitation in the search space [36]. By analyzing solution quality and proximity to optimality, FDB aids in assessing algorithmic convergence and reducing the risk of premature stagnation [37]. Moreover, FDB significantly contributes to advancing optimization methods by enhancing global exploration and local refinement [38].

Numerous studies have leveraged FDB strategies to enhance the capabilities of metaheuristic algorithms. These works demonstrate how FDB facilitates a equilibrium between exploration and exploitation, mitigates premature convergence, and improves the efficiency of solving complex optimization challenges. For example, the Stochastic Fractal Search (SFS) algorithm, despite its solid theoretical foundation, encounters difficulties in maintaining this balance, a common issue among metaheuristic methods. To address this, Sefa et al. [39] developed an FDB operator that effectively emulates fractal patterns to boost diversity. This enhancement led to the creation of the FDB-based SFS algorithm (FDBSFS), which achieved significant performance improvements. Evaluations involving 39 algorithms, 89 unconstrained functions, and 5 constrained problems validated its superiority using statistical analyses like the Wilcoxon signed rank and Friedman tests. FDB has also been applied to tackle the security-constrained optimal power flow (SCOPF) problem, a large-scale NP-hard challenge complicated by renewable energy integration, such as wind and solar, which introduces additional uncertainties. Ugur et al. [40] addressed these complexities by developing an Adaptive Guided Differential Evolution (AGDE) algorithm incorporating FDB. This approach enhanced exploration, exploitation, and diversity, with validation on an IEEE 30-bus system showcasing its superior performance compared to other methods. Similarly, the Artificial Rabbits Optimization (ARO) algorithm benefits from FDB improvements. Ugur [41] introduced roulette and dynamic FDB strategies, resulting in three FDBARO variants. Among these, FDBARO-3, employing the dynamic FDB approach, excelled on CEC 2017 and CEC 2020 benchmarks and demonstrated its capability in solving optimal power flow problems involving renewable energy and FACTS devices. In the context of unmanned aerial vehicles (UAVs), where low-altitude trajectory planning is essential for applications like agriculture, public safety, and defense, Zhu et al. [42] proposed the Multi-strategy Exponential Noise Particle Swarm Optimization (MEPSO) algorithm. By integrating Selective Opposition (SO), exponential noise, and FDB, MEPSO improved flexibility and reduced premature convergence, outperforming other algorithms in delivering safer and more stable flight paths under diverse conditions. For photovoltaic (PV) models, precise parameter estimation is crucial for efficient plant operations. Liu et al. [43] developed an Adaptive Guidance Differential Evolution (AGDE) algorithm combining FDB with Opposition-Based Learning (OBL), which enhanced local search and population diversity. Tests on two PV cells and three PV modules demonstrated that FDB-AGDE outperformed existing methods in accuracy and robustness, offering an effective solution for PV model parameter estimation.

The above studies are summarized in Table 2.

Table 2
Comparison of Studies Using Fitness-Distance Balance (FDB)

Study	Optimization Method	Application	Key Contributions	Limitations
Sefa et al. (2020) [39]	FDB-enhanced SFS	General unconstrained and constrained optimization	Developed FDBSFS with improved exploration-exploitation balance, achieving the highest rank among 39 algorithms tested.	Focused on benchmark functions; limited testing on real-world problems.
Ugur et al. (2021) [40]	FDB-enhanced AGDE	Security-Constrained Optimal Power Flow (SCOPF)	Improved AGDE with FDB for better diversity and search capabilities, outperforming traditional methods in SCOPF optimization.	Application limited to SCOPF problems under specific conditions.
Bakır (2024) [41]	FDB-enhanced ARO	Global optimization and OPF	Introduced RFDB and dFDB strategies, demonstrating superior performance in path planning and OPF problems with renewable energy sources.	Complex implementation with high computational cost.
Zhu et al. (2023) [42]	MEPSO with FDB	UAV low-altitude trajectory planning	Enhanced swarm intelligence algorithm for safe and stable UAV navigation in complex environments.	Focused on UAV-specific applications, limiting generalizability to other domains.
Liu et al. (2024) [43]	FDB-AGDE	Photovoltaic (PV) parameter estimation	Achieved higher accuracy and robustness in PV model parameter estimation by improving AGDE with FDB and OBL.	Focused on PV-specific parameter estimation; limited exploration of other energy systems.

2.2. Competitive Swarm Optimizer (CSO)

The Competitive Swarm Optimizer (CSO) is a swarm intelligence algorithm introduced by Cheng and Jin in 2015, inspired by competitive behaviors in nature [20]. The algorithm operates by iteratively selecting pairs of individuals from a population to compete. The winner retains its position, while the loser moves closer to the winner, updating its position using Eqs. (1) and Eqs. (2), with added randomness to maintain diversity. This mechanism enables CSO to achieve a proper equilibrium between exploration and exploitation, rendering it highly adaptable for diverse optimization challenges, including function optimization, parameter tuning, and path planning. Its simplicity, robustness, and adaptability have driven its successful application across various domains, though further enhancements may be required for tackling complex problems.

$$V_{l,k}(t+1) = R_1(k,t)V_{l,k}(t) + R_2(k,t)(X_{w,k}(t) - X_{l,k}(t)) + \varphi R_3(k,t)(\bar{X}_k(t) - X_{l,k}(t)) \tag{1}$$

In the k-th competition round of generation t, denoting the positions and velocities of the winner and loser be represented as $X_{w,k}(t)$, $X_{l,k}(t)$, and $V_{w,k}(t)$, $V_{l,k}(t)$, respectively, where $k = 1, 2, \ldots, m/2$.

$$X_{l,k}(t+1) = X_{l,k}(t) + V_{l,k}(t+1) \tag{2}$$

3. Our proposal: Dynamic Fitness-Distance Balance Competitive Swarm Optimizer(DFDBCSO)

The development of DFDBCSO addresses the challenges faced by the original Competitive Swarm Optimizer (CSO) in balancing exploration and exploitation. To overcome these limitations, DFDBCSO introduces a Dynamic Fitness-Distance Balance (FDB) scoring mechanism, which adjusts the selection process by incorporating individual performance metrics across various phases of the optimization process. During the initial exploration phase, the FDB score prioritizes the spatial distance between particles and P_{best} (the best solution), fostering competition between active and inactive individuals to maintain population diversity. As the process progresses, the focus gradually shifts to fitness values, promoting competition that encourages weaker individuals to enhance their performance. This adaptive mechanism improves diversity, mitigates premature convergence, and boosts global search effectiveness, making DFDBCSO a robust solution for challenging optimization tasks. The algorithm initializes particles with random positions and velocities within defined boundaries and evaluates their fitness dynamically.

The computation of the FDB score for each particle is defined as follows:

$$\mathrm{FDB}_i = (1-\alpha) \cdot \frac{\|X_i - P_{best}^i\|}{\max_j \|X_j - P_{best}^j\|} + \alpha \cdot \frac{\mathrm{Fitness}_i}{\max_j \mathrm{Fitness}_j}, \tag{3}$$

where:

- $\alpha = \frac{\text{Current Iteration}}{\text{Max Iterations}}$, a dynamic parameter that gradually shifts from emphasizing distance in the early stages to fitness in the later stages.
- $\|X_i - P^i_{best}\|$ represents the Euclidean distance between particle i's current position X_i and its personal best P^i_{best}, normalized by the maximum distance in the population.
- Fitness_i is the fitness value of particle i, normalized by the maximum fitness value in the population.

The dynamic nature of α ensures a smooth transition between exploration and exploitation. At the beginning of the optimization process ($\alpha \approx 0$), the FDB score is primarily determined by the spatial distance, encouraging broad exploration of the search space. As iterations progress, α increases linearly toward 1, causing the FDB score to rely more on fitness values. This shift strengthens exploitation by refining solutions and accelerating convergence toward the global optimum. Since α is directly proportional to the optimization progress, it does not require additional tuning and remains problem-independent, ensuring adaptability across different problem landscapes.

Particles are ranked by their FDB scores, with high and low scorers paired for competition. In each pair, the particle with superior fitness is identified as the winner, while the loser adjusts its velocity according to Eqs. (1). This updated velocity is then used to determine the loser's new position, as detailed in Eqs. (2). Boundary constraints are handled through a mirroring strategy to ensure all particles remain within valid search space boundaries. P_{best} (the best particle) and fitness values are updated whenever an improved solution is discovered. The algorithm iteratively evaluates FDB scores, pairs particles, and updates their positions and velocities until reaching the predefined iteration or evaluation limit. By dynamically adjusting the balance between exploration and exploitation through α, DFDBCSO enhances search efficiency while maintaining population diversity. This allows it to adapt to different optimization problems without requiring problem-specific tuning. Through this mechanism, DFDBCSO proves to be highly effective for solving complex optimization tasks. Figure 1 provides a schematic overview of DFDBCSO. Figure 2 shows its flowchart, while Algorithm 1 outlines the pseudo-code.

Algorithm 1: Dynamic Fitness-Distance Balance Competitive Swarm Optimizer (DFDBCSO)

Input: Population size N, Dimension D, Maximum iterations T
Output: Best solution found X_{best} and its fitness f_{best}

$t \leftarrow 0$, $P(0) \leftarrow$ **initialize**(N, D);
$V(0) \leftarrow$ **initialize_velocity**(N, D);
Evaluate fitness $f(P(0))$ and set P_{best} and $f_{P_{best}}$;
while $t < T$ **do**
    Calculate FDB scores for all particles using Eqs. (3);
    Rank particles based on FDB scores;
    Divide particles into high-score and low-score groups;
    $P(t+1) \leftarrow \emptyset$;
    foreach *pair* (X_H, X_L) *of high and low score particles* **do**
        if $f(X_h) \leq f(X_l)$ **then**
            $X_w \leftarrow X_H$, $X_l \leftarrow X_L$;
        else
            $X_w \leftarrow X_L$, $X_l \leftarrow X_H$;
        end
        Keep X_w in $P(t+1)$;
        Update $V(X_l)$ using Eqs. (1). Update X_l using Eqs. (2);
        Add the updated $X_l(t+1)$ to $P(t+1)$;
    end
    Update P_{best} and $f_{P_{best}}$ if any particle achieves a better fitness;
    $t \leftarrow t+1$;
end
$X_{best} \leftarrow \text{argmin}(f(P(t)))$;
$f_{best} \leftarrow f(X_{best})$;
return X_{best}, f_{best};

4. Numerical Experiments in CEC benchmarks

This section details numerical experiments on the CEC benchmarks to assess the DFDBCSO's performance. Section 4.1 describes the experimental setup, and Section 4.2 analyzes the results from CEC2020 and CEC2022.

4.1. Experimental Settings

4.1.1. Experimental Environments

- Device: MacBook Pro 2017
- Operating System: macOS Ventura 13.6.6
- CPU: 2.3GHz Intel Core i5
- GPU: Intel Iris Plus Graphics 640 1536MB
- RAM: 16 GB 2133 MHz LPDDR3
- Programming Language: Python 3.11.7

The selected hardware and software configurations ensure efficient experiment execution. Moreover, sharing these details promotes reproducibility, enabling other researchers to obtain similar outcomes under comparable conditions.

4.1.2. Benchmark Functions

This section introduces the benchmarks used in the study, outlined as follows:

- CEC2020: Comprising 10 functions, this benchmark includes unimodal, multimodal, hybrid, and composite types.
- CEC2022: Featuring 12 functions, this benchmark also covers unimodal, multimodal, hybrid, and composite categories.

Both CEC benchmarks are obtained from the Opfunu library [44].

4.1.3. Competitor Algorithms and Parameters

This section outlines the competitor algorithms and their parameter settings. For comparison, we selected four categories of optimization methods:

- DE-based optimizers: DE, JADE, L-SHADE, jSO.
- PSO-based optimizers: PSO, CSO.
- Evolutionary-based optimizers: CMA-ES.
- CSO variants: FDBCSO (without dynamic parameter α), MCSO, SCSO, WLCSO.

DE, JADE, L-SHADE, PSO, CSO, and CMA-ES are implemented using the MEALPY library [45–47], while jSO is sourced from the PyADE library[1]. MCSO, SCSO, and WLCSO are developed based on descriptions provided in their respective authors' original publications. The maximum number of fitness evaluations (FEs) is set to 1000 $\times$ D, where D represents the dimension size, Table 3 provides a summary of the parameter configurations for these algorithms.

All comparison algorithms were set up using the recommended parameters from their original references. To ensure reliability and mitigate the effects of randomness, each algorithm was executed 20 times for each optimization problem.

4.2. Comparison Experiments Results and Analysis in CEC benchmarks

This section presents the comparative experiment results on the CEC benchmarks, accompanied by a comprehensive statistical analysis. The Holm Multiple Comparison Test (HMT) is applied, using the proposed DFDBCSO method as the baseline. The symbols $+$, $\approx$, and $-$ indicate whether DFDBCSO significantly outperforms, shows no significant difference, or underperforms compared to the competing algorithms. The best average fitness values are highlighted in bold.

Table 4 provides the statistical results and average rankings of the optimizers for the CEC2017, CEC2020 and CEC2022 benchmarks, with detailed experimental data included in Appendices B and C.

[1] https://github.com/xKuZz/pyade

Table 3
Parameter settings of algorithms

MAs	Parameters	Value
DE	N	100
	mutation scheme	DE/cur-to-rand/1
	F and Cr	0.8 and 0.8
PSO	N	100
	w, c_1, and c_2	1, 2.05, and 2.05
	v_{max} and v_{min}	2 and -2
CMA-ES	N	100
	σ	1.3
JADE	N	100
	μ_F and μ_{Cr}	0.5 and 0.5
	σ_F and σ_{Cr}	0.1 and 0.1
L-SHADE	N_{max} and N_{min}	18 × D and 4
	μ_F and μ_{Cr}	0.5 and 0.5
	σ_F and σ_{Cr}	0.1 and 0.1
	memory size H	100
CSO	N	400
	ϕ	0.1
jSO	N_{max} and N_{min}	12 × D and 4
	p_{max} and p_{min}	0.25 and 0.125
	$\boldsymbol{M}_F$	0.3
MCSO	N	400
	ϕ_1	0.1
	ϕ_2	0.1
SCSO	N	200
	w	1.8
WLCSO	N	200
	ϕ	0.1
FDBCSO	N	200
	α	0.5
	ϕ	0.1
DFDBCSO	N	200
	ϕ	0.1

Figures 3, 4, 5, 6 depict the convergence curves, box plots, radar rankings, and average rankings for selected functions of the CEC2017 benchmark in 30 dimensions, Figures 7, 8, 9, 10 show similar visualizations for 50 dimensions, and Figures 11, 12, 13, 14 show similar visualizations for 100 dimensions.

Figures 15, 16, 17, 18 depict the convergence curves, box plots, radar rankings, and average rankings for selected functions of the CEC2020 benchmark in 10 dimensions, Figures 19, 20, 21, 22 show similar visualizations for 20 dimensions, and Figures 23, 24, 25, 26 show similar visualizations for 100 dimensions.

For the CEC2022 benchmark, Figures 27, 28, 29, 30 illustrate these metrics for 10-dimensional functions, and Figures 31, 32, 33, 34 provide corresponding results for 20-dimensional functions.

Table 4
The statistical results and average ranks of Comparison Experiments in CEC benchmarks

Bench.	Dim.		DE	PSO	CMA-ES	JADE	L-SHADE	CSO	jSO	MCSO	SCSO	WLCSO	FDBCSO	DFDBCSO
CEC2017	30	+/≈/−:	27/1/1	29/0/0	24/1/4	11/7/11	18/5/6	18/8/3	20/4/5	20/5/4	19/6/4	27/1/1	14/13/2	-
		Avg. ranks:	10.79	11.03	7.83	3.93	5.21	4.59	6.90	6.10	5.79	9.24	3.41	**3.17**
	50	+/≈/−:	29/0/0	29/0/0	24/0/5	13/1/15	15/6/8	10/10/9	22/1/6	19/3/7	19/4/6	28/0/1	5/23/1	-
		Avg. ranks:	11.48	11.03	7.93	**3.55**	5.10	3.93	6.52	5.79	5.86	9.14	3.83	3.83
	100	+/≈/−:	29/0/0	29/0/0	26/1/2	13/5/11	13/7/9	6/9/14	21/2/6	25/0/4	24/1/4	27/1/1	14/14/1	-
		Avg. ranks:	11.72	10.97	8.24	3.33	4.62	**2.72**	5.83	7.24	7.21	8.45	3.72	3.52
CEC2020	10	+/≈/−:	8/2/0	10/0/0	7/1/2	7/2/1	6/1/3	9/1/0	8/2/0	7/2/1	7/0/3	9/1/0	5/4/1	-
		Avg. ranks:	9.90	10.70	6.50	5.90	3.50	7.20	8.90	4.10	5.50	9.20	3.60	**3.00**
	20	+/≈/−:	9/0/1	10/0/0	8/1/1	9/0/1	5/3/2	8/2/0	7/1/2	8/1/1	7/2/1	9/0/1	2/7/1	-
		Avg. ranks:	10.50	10.90	7.70	5.60	3.70	5.60	8.10	5.00	5.60	9.40	**2.90**	3.00
	100	+/≈/−:	10/0/0	10/0/0	10/0/0	8/0/2	5/2/3	4/2/4	9/0/1	10/0/0	10/0/0	9/0/1	8/2/0	-
		Avg. ranks:	11.70	10.60	8.70	4.70	3.40	3.00	5.80	7.40	9.50	7.90	2.90	**2.40**
CEC2022	10	+/≈/−:	12/0/0	12/0/0	10/0/2	12/0/0	10/1/1	11/1/0	9/3/0	9/2/1	11/1/0	11/1/0	6/6/0	-
		Avg. ranks:	10.67	10.92	6.17	6.67	4.17	6.67	9.50	4.00	5.42	9.25	2.58	**2.00**
	20	+/≈/−:	12/0/0	12/0/0	10/1/1	11/1/0	11/0/1	10/2/0	8/2/2	11/1/0	8/3/1	12/0/0	2/10/0	-
		Avg. ranks:	11.08	11.00	7.58	7.00	4.58	5.00	6.17	6.33	4.92	9.58	**2.25**	2.50

Table 5
Results of optimizers in CBDP.

MAs	mean	std	best	worst
DE	1.181888e+04 +	3.346725e+01	1.181120e+04	1.196476e+04
PSO	1.270984e+04 +	3.295995e+02	1.206164e+04	1.365640e+04
CMA-ES	1.181481e+04 +	1.574554e+01	1.181120e+04	1.188345e+04
JADE	1.181252e+04 +	1.042038e+00	1.181476e+04	1.181884e+04
L-SHADE	1.181536e+04 +	1.119790e+01	1.181120e+04	1.185597e+04
CSO	1.181664e+04 +	1.042038e+00	1.181476e+04	1.181884e+04
jSO	1.277960e+04 +	3.423908e+02	1.211853e+04	1.329435e+04
MCSO	1.224564e+04 +	9.591808e+01	1.208664e+04	1.243799e+04
SCSO	1.181126e+04 +	1.805925e-02	1.181123e+04	1.181130e+04
WLCSO	1.181988e+04 +	3.990626e+00	1.181356e+04	1.183164e+04
DFDBCSO	**1.181120e+04**	5.688436e-04	1.181120e+04	1.181120e+04

5. Numerical Experiments on Engineering Problems

This section presents a comprehensive analysis of the experimental results and assesses the performance of the optimizers across five engineering problems.

5.1. Cantilever Beam Design Problem (CBDP)

The cantilever beam design is a well-known research topic in engineering optimization, focusing on identifying five optimal structural variables to minimize total weight while meeting the required load capacity. Figure 35 illustrates the cantilever beam's structure, which consists of several hollow square sections. The mathematical formulation of the problem can refer to Appendix D.1. Table 5 presents the experimental outcomes of each algorithm for CBDP. Figure 36 illustrates the rankings, convergence trends, and boxplot of the optimizers for CBDP.

The HMT results in Table 5, using DFDBCSO as the baseline, show that it outperformed other algorithms 10 times +, with no ≈ or −. Furthermore, DFDBCSO ranked first in the average rankings. These results highlight the exceptional performance of DFDBCSO on CBDP problems compared to other algorithms.

5.2. Rolling Element Bearing Design Problem (REBDP)

The second constrained engineering optimization problem involves designing a rolling element bearing to maximize its dynamic load-carrying capacity (C_d) while meeting specified constraints and parameters. Figure 37

Table 6
Results of optimizers in REBDP.

MAs	mean	std	best	worst
DE	1.472078e+04 ≈	2.120493e+02	1.461717e+04	1.544902e+04
PSO	2.039329e+04 +	3.213298e+03	1.577903e+04	2.870501e+04
CMA-ES	1.840231e+04 +	4.504983e+03	1.472691e+04	3.077683e+04
JADE	1.772075e+04 +	9.804297e+01	1.494091e+04	1.527336e+04
L-SHADE	1.596513e+04 +	1.006024e+03	1.481064e+04	1.829744e+04
CSO	1.506311e+04 +	9.804297e+01	1.494091e+04	1.527336e+04
jSO	2.022598e+04 +	2.298679e+03	1.625597e+04	2.557731e+04
MCSO	1.628890e+04 +	3.360230e+02	1.572674e+04	1.698845e+04
SCSO	1.471804e+04 +	2.071041e+01	1.467340e+04	1.475710e+04
WLCSO	1.497074e+04 +	8.401772e+01	1.481855e+04	1.514547e+04
DFDBCSO	**1.464761e+04**	1.351276e+01	1.463239e+04	1.467867e+04

Table 7
Results of optimizers in SRDP.

MAs	mean	std	best	worst
DE	**2.769573e+03** −	5.865796e-01	2.769323e+03	2.771972e+03
PSO	2.926492e+03 +	4.645288e+01	2.865093e+03	3.054709e+03
CMA-ES	2.906965e+03 +	3.448212e+02	2.777204e+03	4.399011e+03
JADE	2.814642e+03 +	3.688791e+00	2.780908e+03	2.797721e+03
L-SHADE	2.793315e+03 +	8.730312e+00	2.779410e+03	2.815221e+03
CSO	2.787973e+03 +	3.688791e+00	2.780908e+03	2.797721e+03
jSO	2.939505e+03 +	6.257824e+01	2.843408e+03	3.118969e+03
MCSO	2.839238e+03 +	1.998590e+01	2.797505e+03	2.873244e+03
SCSO	2.774456e+03 +	2.613691e+00	2.770509e+03	2.782435e+03
WLCSO	2.788621e+03 +	3.794402e+00	2.781349e+03	2.796118e+03
DFDBCSO	2.769632e+03	1.233934e-01	2.769474e+03	2.769898e+03

illustrates the bearing design problem, with its mathematical formulation provided in Appendix D.2. Table 6 presents the experimental results for each algorithm on REBDP, and Figure 38 displays the rankings, convergence trends, and boxplots of the optimizers.

Using DFDBCSO as the baseline, the HMT results in Table 6 indicate 9 occurrences of +, 1 occurrence of ≈, and 0 occurrences of −. Additionally, DFDBCSO achieves the highest average ranking. These results demonstrate DFDBCSO's superior performance in the REBDP problem.

5.3. Speed Reducer Design Problem (SRDP)

This optimization task focuses on minimizing the mass of a speed reducer by adjusting seven design parameters: face width (z_1), module of teeth (z_2), number of pinion teeth (z_3), shaft lengths (z_4, z_5), and shaft diameters (z_6, z_7). The design must comply with constraints related to gear bending stress, shaft deflection, surface stress, and shaft stress. Figure 39 illustrates the speed reducer design, while the corresponding mathematical formulation is detailed in Appendix D.3. Table 7 presents the experimental outcomes for SRDP, and Figure 40 depicts the rankings, convergence patterns, and boxplots of the optimizers.

Using DFDBCSO as the baseline, the HMT results in Table 7 reveal 9 times of +, 0 times of ≈, and 1 time of −. Additionally, DFDBCSO achieved an average ranking of second, following DE. These experimental findings highlight that while DFDBCSO ranked second overall, it maintains a notable advantage over all algorithms except DE.

Table 8
Results of optimizers in TCSDP.

MAs	mean	std	best	worst
DE	2.161512e+07 +	6.245821e+06	1.907723e+07	4.375031e+07
PSO	2.051055e+07 +	8.734700e+05	1.914763e+07	2.232365e+07
CMA-ES	3.533231e+07 +	1.165891e+07	2.091565e+07	7.075649e+07
JADE	2.248836e+07 +	3.903724e+04	1.884803e+07	1.901409e+07
L-SHADE	2.019158e+07 +	8.041284e+05	1.943854e+07	2.267457e+07
CSO	1.893577e+07 +	3.903724e+04	1.884803e+07	1.901409e+07
jSO	2.035681e+07 +	6.971338e+05	1.909005e+07	2.209162e+07
MCSO	1.914814e+07 +	1.694608e+05	1.889657e+07	1.945045e+07
SCSO	**1.879059e+07** −	1.622701e+04	1.876197e+07	1.882026e+07
WLCSO	1.879509e+07 −	2.324616e+04	1.876299e+07	1.883977e+07
DFDBCSO	1.888769e+07	4.383316e+04	1.879515e+07	1.896211e+07

Table 9
Results of optimizers in WBDP.

MAs	mean	std	best	worst
DE	2.148373e+00 +	2.201145e-01	1.698769e+00	2.605313e+00
PSO	1.899399e+00 +	1.173166e-01	1.653739e+00	2.130084e+00
CMA-ES	3.067444e+00 +	6.253772e-01	2.172170e+00	4.383420e+00
JADE	2.367992e+00 +	5.084546e-02	1.617414e+00	1.804231e+00
L-SHADE	2.123273e+00 +	2.104021e-01	1.702986e+00	2.662079e+00
CSO	1.673493e+00 +	5.084546e-02	1.617414e+00	1.804231e+00
jSO	1.901665e+00 +	1.398740e-01	1.689803e+00	2.170348e+00
MCSO	1.629343e+00 +	2.130129e-02	1.591284e+00	1.683098e+00
SCSO	1.617641e+00 +	9.548686e-03	1.602558e+00	1.638251e+00
WLCSO	1.632174e+00 +	1.856955e-02	1.597102e+00	1.660593e+00
DFDBCSO	1.599867e+00	5.274004e-02	1.581161e+00	1.821193e+00

5.4. Tension/compression Spring Design Problem (TCSDP)

This optimization task aims to minimize the weight of a tension/compression spring by adjusting three key variables: wire diameter (d), average coil diameter (D), and the number of active coils (N). The design must adhere to constraints involving shear stress, surge frequency, and deflection limits. Figure 41 depicts the spring configuration, with the mathematical formulation detailed in Appendix D.4. Table 8 presents the experimental results for TCSDP, and Figure 42 showcases the rankings, convergence trends, and boxplots of the optimizers.

Using DFDBCSO as the baseline, the HMT results in Table 8 indicate 8 times of +, 0 of ≈, and 2 of −. Overall, DFDBCSO ranks third, demonstrating a clear advantage over all algorithms except SCSO and WLCSO.

5.5. Welded Beam Design Problem (WBDP)

This task aims to reduce the fabrication cost of a welded beam while adhering to constraints on shear stress, bending stress, buckling load, and deflection limits. The design involves four primary variables: weld thickness (h), joint length (l), beam height (t), and beam thickness (b). Figure 43 illustrates the welded beam configuration, with its mathematical formulation detailed in Appendix D.5. Table 9 presents the experimental results for WBDP, and Figure 44 visualizes the rankings, convergence trends, and boxplots of the optimizers.

The HMT results for DFDBCSO, presented in Table reftbl:5.5.1, reveal that the + appears 10 times, while ≈ and − appear 0 times each. Additionally, DFDBCSO achieves the highest average ranking, underscoring its superior performance on WBDP problems compared to other algorithms.

Table 10
Comparison of Experimental Results with Regression Models for Predicting Superconductor Critical Temperatures..

Models	MSE	MAE	R^2
DFDBCSO-Ensemble	**97.65**	**5.62**	**0.92**
ExtraTrees	99.47	5.66	0.91
GradientBoosting	101.97	5.78	0.91
RandomForest	102.55	5.68	0.91
KNN	154.88	7.12	0.87
DecisionTree	182.02	7.95	0.84
SVR	182.15	7.55	0.84
Ridge	317.76	13.28	0.72
Lasso	320.46	13.36	0.72
AdaBoost	376.89	16.13	0.67

6. Application of DFDBCSO-Ensemble in Superconductor Critical Temperature Prediction

This section introduces the DFDBCSO-Ensemble, developed for predicting the critical temperature of superconductors. Its framework is shown in Figure 45. Initially, nine regression models were employed for temperature prediction, and the three most accurate models were selected to construct the DFDBCSO-Ensemble. The DFDBCSO algorithm dynamically adjusts the weights of these models, improving the ensemble's overall prediction accuracy.

This study leverages a publicly available dataset from Kaggle[2], which includes 81 features derived from 21,263 superconductors, with critical temperature recorded in column 82. The dataset is split into 70% for training, 15% for validation, and 15% for testing. The DFDBCSO algorithm utilizes a population size of 200 and runs for up to 500 iterations. Table 10 summarizes the results achieved by the DFDBCSO-Ensemble method. Figure 46 shows the Actual vs. Predicted plot, while Figure 47 illustrates the model's performance in forecasting superconductor critical temperatures.

7. Discussion

7.1. Theoretical Analysis of the DFDB Strategy

While the empirical validation of DFDBCSO has demonstrated its superior performance, it is essential to analyze the theoretical underpinnings that contribute to its success. The Dynamic Fitness-Distance Balance (DFDB) strategy enhances the Competitive Swarm Optimizer (CSO) by introducing an adaptive competition mechanism that dynamically regulates the balance between exploration and exploitation. From a theoretical perspective, DFDB effectively prevents premature convergence by maintaining population diversity in the early optimization stages while accelerating convergence in later stages. The balance between exploration and exploitation is controlled through the Fitness-Distance Balance (FDB) score, which integrates two critical factors: the relative fitness difference and the spatial distribution of individuals. By dynamically adjusting these two factors, DFDB shifts the algorithm's focus adaptively. During the early optimization phase, the distance component plays a dominant role, encouraging particles to explore diverse regions of the search space. As optimization progresses, fitness-based selection becomes more dominant, promoting convergence toward optimal solutions.

One of the key reasons for DFDBCSO's effectiveness on multimodal and hybrid functions is its ability to maintain exploration diversity while refining search directions. Standard CSO lacks an adaptive mechanism to adjust search intensity based on landscape complexity, which often leads to premature convergence in multimodal landscapes. DFDBCSO, in contrast, incorporates adaptive exploration and controlled exploitation, ensuring that solutions escape local optima effectively. Moreover, the controlled transition between exploration and exploitation ensures that the algorithm satisfies conditions necessary for global convergence, such as maintaining sufficient diversity while improving search accuracy. These characteristics make DFDBCSO particularly effective for handling complex, high-dimensional, and dynamic optimization landscapes.

[2] https://www.kaggle.com/datasets/munumbutt/superconductor-dataset

7.2. Comparison Experiments Discussion in CEC Benchmarks

According to the experiments results, DFDBCSO demonstrates strong performance in the CEC benchmarks. For example, In the 10-dimensional of CEC2022 experiment, DFDBCSO outperformed other algorithms 107 times (+), showed no significant difference 18 times (≈), and was outperformed only 7 times (−), highlighting its competitive edge. Similarly, in the 20-dimensional of CEC2022 experiment, DFDBCSO achieved + in 107 cases, ≈ in 20 cases, and − in only 5 cases, further confirming its effectiveness. A deeper examination of function-wise performance reveals that DFDBCSO consistently excels on multimodal and hybrid functions, such as f_1, f_2, f_6, and f_8. These functions often exhibit numerous local optima, making it challenging for traditional algorithms to escape suboptimal regions. DFDBCSO's ability to dynamically adjust the exploration-exploitation balance allows it to navigate these landscapes more efficiently.

The results indicate that DFDBCSO maintains higher robustness and adaptability across diverse problem types, particularly in cases where maintaining diversity is crucial. Unlike many evolutionary and swarm intelligence algorithms that struggle with premature convergence, DFDBCSO dynamically regulates the selection pressure based on function complexity. By integrating fitness and distance metrics, the DFDB strategy enhances search efficiency, ensuring that particles maintain a diverse yet structured search process. This adaptability allows DFDBCSO to outperform standard algorithms in highly complex problem spaces while retaining stability across different function classes. The consistent high performance across multiple benchmarks reinforces DFDBCSO's suitability as a robust optimization approach.

7.3. Comparison Experiments Discussion on Engineering Problems

DFDBCSO was evaluated on five engineering design problems to assess its effectiveness in solving real-world optimization challenges. The results demonstrate DFDBCSO's strong capability in handling constrained and highly nonlinear optimization tasks. It ranked first in the Cantilever Beam Design Problem (CBDP), Rolling Element Bearing Design Problem (REBDP), and Welded Beam Design Problem (WBDP), indicating its effectiveness in structural and material optimization. Additionally, it secured second place in the Speed Reducer Design Problem (SRDP) and third in the Tension/Compression Spring Design Problem (TCSDP), showcasing its strong performance in mechanical design optimization.

The high efficiency of DFDBCSO in these problems can be attributed to its adaptive search strategy, which allows the algorithm to balance global exploration with fine-tuned local exploitation, leading to more precise solutions under real-world constraints. Engineering optimization problems often require algorithms to handle highly constrained spaces where feasible regions are limited, and traditional optimizers struggle to find optimal designs within these constraints. DFDBCSO's ability to dynamically regulate its search behavior based on constraint severity enables it to outperform conventional techniques in engineering applications. Its robust adaptability makes it a valuable tool for structural, mechanical, and industrial design optimization tasks.

7.4. Comparison Experiments Discussion in DFDBCSO-Ensemble for Superconductor Critical Temperature Prediction

The DFDBCSO-Ensemble achieved significant improvements over the second-ranked model, ExtraTrees, demonstrating its superior predictive capability. It reduced the Mean Squared Error (MSE) by 1.82, decreased the Mean Absolute Error (MAE) by 0.04, and improved the R-squared score (R^2) by 0.01, leading to more precise and reliable estimations of superconducting critical temperatures. These results highlight the exceptional effectiveness and competitiveness of the DFDBCSO-Ensemble, which seamlessly integrates machine learning models with the adaptive optimization capabilities of DFDBCSO.

By leveraging intelligent weight optimization, the ensemble dynamically adjusts the contribution of each base model, assigning higher weights to more accurate predictions while mitigating the impact of less reliable models in real time. This adaptive fusion strategy enhances the ensemble's robustness, ensuring consistently high performance across varying datasets and conditions. Unlike static ensemble learning techniques, DFDBCSO-Ensemble continuously adapts its internal weighting mechanisms, making it particularly useful for high-variability scientific applications where model uncertainty must be actively managed. Furthermore, DFDBCSO-Ensemble provides a generalized framework for regression problems, demonstrating its potential to be extended beyond superconducting temperature prediction to other scientific and engineering applications where high-precision modeling is critical. These findings reinforce the importance of optimization-driven machine learning in advancing data-driven predictive modeling.

7.5. Discussion on Computational Complexity

The computational complexity of DFDBCSO is primarily determined by the competition-based selection process and the dynamic adjustment of search behavior. Unlike traditional CSO, where pairwise competition is conducted with a fixed strategy, DFDBCSO introduces an adaptive fitness-distance balance mechanism, requiring additional computations to evaluate both fitness and spatial diversity. This results in a per-iteration complexity of $O(N^2)$, where N is the population size, as it involves distance calculations and adaptive competition adjustments. Compared to standard CSO, which operates at $O(N)$ complexity due to its simpler selection mechanism, DFDBCSO incurs additional overhead. However, in practice, this additional cost is mitigated by its faster convergence, as the adaptive competition strategy reduces unnecessary iterations required for global search refinement. When compared to differential evolution-based algorithms like L-SHADE and jSO (typically $O(N \log N)$ due to their sorting-based mutation strategies), DFDBCSO remains computationally competitive while achieving better solution quality on complex benchmarks. Experimental results suggest that the additional complexity introduced by DFDB does not significantly impact runtime in practical applications, as its dynamic adaptation accelerates convergence while maintaining high solution accuracy.

8. Conclusion

The original CSO algorithm lacks a dedicated individual selection mechanism, limiting its ability to dynamically adjust exploration between the early and late optimization phases. To address this limitation, we introduce the Dynamic Fitness-Distance Balance (DFDB) strategy, which refines the exploration-exploitation process, enabling a more effective search throughout the optimization.

In the initial exploration phase, the DFDB strategy prioritizes the Fitness-Distance Balance (FDB) score, integrating distance metrics between individuals and P_{best}. This approach fosters diverse interactions by encouraging comparisons between highly active individuals (long distances) and less active ones (short distances), allowing the algorithm to explore a broader solution space and mitigate premature convergence. As optimization progresses, the FDB score gradually shifts focus toward fitness-based selection, where individuals with stronger fitness influence weaker ones, leading to adaptive improvements. This targeted refinement enhances the algorithm's efficiency in converging toward optimal solutions while maintaining diversity.

By dynamically adapting the balance between exploration and exploitation, DFDBCSO significantly enhances CSO's performance. It maintains population diversity in early iterations and focuses on precise convergence in later stages, making it a robust optimization approach for complex search spaces.

Extensive experiments on the CEC2017, CEC2020 and CEC2022 benchmarks demonstrate that DFDBCSO consistently achieves top performance across various dimensions, outperforming competing algorithms in both accuracy and robustness. Additionally, DFDBCSO was evaluated on five engineering optimization problems, ranking first in three cases (CBDP, REBDP, and WBDP), second in SRDP, and third in TCSDP. These results confirm its strong adaptability and effectiveness in real-world engineering applications.

Furthermore, we developed DFDBCSO-Ensemble, a weight-based ensemble model that applies DFDBCSO for superconductor critical temperature prediction. By dynamically optimizing model weights, DFDBCSO-Ensemble achieves superior accuracy across multiple evaluation metrics, surpassing traditional regression models. This highlights its potential for addressing complex scientific and engineering prediction tasks.

Building on the success of DFDBCSO, future research will focus on extending its capabilities to multi-objective and constrained optimization problems. A key enhancement will involve integrating adaptive constraint-handling techniques that dynamically adjust the balance between feasible and infeasible solutions, improving performance in real-world applications with strict design constraints. Additionally, we aim to explore multi-objective DFDBCSO, where the DFDB strategy is adapted to balance multiple conflicting objectives, ensuring diversity and convergence in Pareto-optimal solutions.

Another research direction involves developing self-adaptive parameter tuning mechanisms, enabling DFDBCSO to automatically adjust its exploration-exploitation trade-off based on problem characteristics. This will enhance its robustness in tackling increasingly complex and high-dimensional problems. Furthermore, we plan to extend DFDBCSO's application to large-scale industrial and societal challenges, including energy management(optimizing resource allocation and scheduling), manufacturing optimization (enhancing production efficiency and process optimization), medical optimization (improving diagnostics and personalized treatment planning), and intelligent

transportation (optimizing traffic flow and route planning). These extensions will further validate DFDBCSO's practical value and expand its potential for solving real-world high-impact optimization tasks.

9. Statement and Declarations

9.1. Competing interest

The authors declare no competing interests.

9.2. Data availability

The source code of this research can be downloaded from https://github.com/soyo123/DFDBCSO/tree/main.

9.3. Acknowledgment

This work was supported by JST SPRING, Grant Number JPMJSP2119 and JSPS KAKENHI Grant Number 21A402 and 24K15098.

References

[1] J. Ji, T. Wu, C. Yang, Neural population dynamics optimization algorithm: A novel brain-inspired meta-heuristic method, Knowledge-Based Systems 300 (2024) 112194.
[2] X. Zhong, B. Liang, Elite-based butterfly optimization algorithm and its application in speckle projection technique, Applied Soft Computing 169 (2024) 112561.
[3] S. Oladejo, S. Ekwe, S. Mirjalili, The hiking optimization algorithm: A novel human-based metaheuristic approach, Knowledge-Based Systems 296 (2024) 111880.
[4] L. Xiao, Z. Zhang, K. Huang, J. Jiang, Y. Peng, Noise optimization in artificial neural networks, IEEE Transactions on Automation Science and Engineering (2024) 1–14.
[5] M. Ghazaan, A. Salmani Oshnari, A. Salmani Oshnari, A novel adaptive optimization scheme for advancing metaheuristics and global optimization, Swarm and Evolutionary Computation 91 (2024) 101779.
[6] T. Hamadneh, B. Batiha, O. Alsayyed, G. Bektemyssova, Z. Montazeri, M. Dehghani, K. Eguchi, On the application of potter optimization algorithm for solving supply chain management application, International Journal of Intelligent Engineering & Systems 17 (5) (2024) 88.
[7] S. Alomari, K. Kaabneh, I. AbuFalahah, S. Gochhait, I. Leonova, Z. Montazeri, M. Dehghani, K. Eguchi, Carpet weaver optimization: A novel simple and effective human-inspired metaheuristic algorithm, International Journal of Intelligent Engineering & Systems 17 (4) (2024) 230.
[8] T. Hamadneh, B. Batiha, F. Werner, Z. Montazeri, M. Dehghani, B. Gulnara, K. Eguchi, Fossa optimization algorithm: A new bio-inspired metaheuristic algorithm for engineering applications, International Journal of Intelligent Engineering & Systems 17 (2024) 1038 – 1047.
[9] T. Hamadneh, K. Kaabneh, O. Alssayed, K. Eguchi, D. S. Gochhait, I. Leonova, M. Dehghani, Addax optimization algorithm: A novel nature-inspired optimizer for solving engineering applications, International Journal of Intelligent Engineering & Systems 17 (2024) 732–743.
[10] S. Omari, K. Kaabneh, I. Abufalahah, K. Eguchi, D. S. Gochhait, I. Leonova, Z. Montazeri, M. Dehghani, Dollmaker optimization algorithm: A novel human-inspired optimizer for solving optimization problems, International Journal of Intelligent Engineering & Systems 17 (2024) 816–828.
[11] T. Hamadneh, B. Batiha, O. Al Baik, B. Gulnara, Z. Montazeri, F. Werner, G. Dhiman, M. Dehghani, K. Eguchi, Sales training based optimization: A new human-inspired metaheuristic approach for supply chain management, International Journal of Intelligent Engineering & Systems (2024) 1325.
[12] T. Hamadneh, B. Batiha, G. Gharib, Z. Montazeri, F. Werner, G. Dhiman, M. Dehghani, R. Jawad, E. Aram, I. Ibraheem, K. Eguchi, Orangutan optimization algorithm: An innovative bio-inspired metaheuristic approach for solving engineering optimization problems, International Journal of Intelligent Engineering & Systems 18 (2024) 2025.
[13] T. Hamadneh, B. Batiha, O. Al Baik, Z. Montazeri, O. Parkash, F. Werner, G. Dhiman, M. Dehghani, K. Eguchi, Spider-tailed horned viper optimization: An effective bio-inspired metaheuristic algorithm for solving engineering applications, International Journal of Intelligent Engineering & Systems 18 (2024) 2025.
[14] Y. Yuan, Q. Shen, W. Xi, S. Wang, J. Ren, J. Yu, Q. Yang, Multidisciplinary design optimization of dynamic positioning system for semi-submersible platform, Ocean Engineering 285 (2023) 115426.
[15] Y. Yuan, Q. Yang, J. Ren, X. Mu, Z. Wang, Q. Shen, Y. Li, Short-term power load forecasting based on skdr hybrid model, Electrical Engineering (2024) 1–17.
[16] Y. Yuan, Q. Yang, J. Ren, X. Mu, Z. Wang, Q. Shen, W. Zhao, Attack-defense strategy assisted osprey optimization algorithm for pemfc parameters identification, Renewable Energy 225 (2024) 120211.
[17] L. Kong, M. B. Jasser, S. Ajibade, A. Wagdy, A systematic review on software reliability prediction via swarm intelligence algorithms, Journal of King Saud University - Computer and Information Sciences 36 (2024) 102132.
[18] G. H. Valencia-Rivera, M. T. Benavides-Robles, A. Vela Morales, I. Amaya, J. M. Cruz-Duarte, J. C. Ortiz-Bayliss, J. G. Avina-Cervantes, A systematic review of metaheuristic algorithms in electric power systems optimization, Applied Soft Computing 150 (2024) 111047.
[19] G. Li, T. Zhang, C.-Y. Tsai, L. Yao, Y. Lu, J. Tang, Review of the metaheuristic algorithms in applications: Visual analysis based on bibliometrics, Expert Systems with Applications 255 (2024) 124857.
[20] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Transactions on Cybernetics 45 (2) (2015) 191–204.

[21] J. Tang, G. Liu, Q. Pan, A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends, IEEE/CAA Journal of Automatica Sinica 8 (10) (2021) 1627–1643.
[22] E. H. Houssein, A. G. Gad, K. Hussain, P. N. Suganthan, Major advances in particle swarm optimization: Theory, analysis, and application, Swarm and Evolutionary Computation 63 (2021) 100868.
[23] J. Nayak, H. Swapnarekha, B. Naik, G. Dhiman, S. Vimal, 25 years of particle swarm optimization: Flourishing voyage of two decades, Archives of Computational Methods in Engineering 30 (3) (2023) 1663–1725.
[24] O. Ertenlice, C. Kalayci, A survey of swarm intelligence for portfolio optimization: Algorithms and applications, Swarm and Evolutionary Computation 39 (02 2018).
[25] H. Jin, X. Wang, S. Cheng, Y. Sun, M. Zhang, H. Lu, H. Wu, Y. Shi, Solving dynamic multimodal optimization problems via a niching-based brain storm optimization with two archives algorithm, Swarm and Evolutionary Computation 89 (2024) 101649.
[26] A. Jaafari, M. Panahi, D. Mafi-Gholami, O. Rahmati, H. Shahabi, A. Shirzadi, S. Lee, D. Bui, B. Pradhan, Swarm intelligence optimization of the group method of data handling using the cuckoo search and whale optimization algorithms to model and predict landslides, Applied Soft Computing 116 (2021) 108254.
[27] W. Li, T. Zhang, R. Wang, S. Huang, J. Liang, Multimodal multi-objective optimization: Comparative study of the state-of-the-art, Swarm and Evolutionary Computation 77 (2023) 101253.
[28] Y. Cao, R. Zhong, J. Yu, M. Munetomo, Optimization of electricity consumption forecasting models via hyper-heuristic algorithm, in: 2024 6th International Conference on Data-driven Optimization of Complex Systems (DOCS), 2024, pp. 114–120.
[29] C.-M. Wen, M. Ierapetritou, Topology-informed derivative-free metaheuristic optimization method, Computers & Chemical Engineering 194 (2024) 108973.
[30] J. Li, X. Li, L. Gao, An operator-inspired framework for metaheuristics and its applications on job-shop scheduling problems, Applied Soft Computing 157 (2024) 111522.
[31] U. Sadana, A. Chenreddy, E. Delage, A. Forel, E. Frejinger, T. Vidal, A survey of contextual optimization methods for decision-making under uncertainty, European Journal of Operational Research 320 (2) (2025) 271–289.
[32] B. Raj, I. Ahmedy, M. Y. I. Idris, R. M. Noor, A hybrid sperm swarm optimization and genetic algorithm for unimodal and multimodal optimization problems, IEEE Access 10 (2022) 109580–109596.
[33] A. Solak, A. Onat, O. Kilinç, An improved bare bones particle swarm optimization algorithm based on sequential update mechanism and a modified structure, IEEE Access 13 (2025) 4789–4814.
[34] J. Liu, B. Liang, W. Ji, An anomaly detection approach based on hybrid differential evolution and k-means clustering in crowd intelligence, International Journal of Crowd Science 5 (2) (2021) 129–142.
[35] H. T. Kahraman, S. Aras, E. Gedikli, Fitness-distance balance (fdb): A new selection method for meta-heuristic search algorithms, Knowledge-Based Systems 190 (11 2019).
[36] Q. Liu, K. Sun, X. Tang, J. Huo, Camera calibration based on lightweight fan-shaped target detection and fitness-distance-balance chaotic marine predators algorithm, Optics & Laser Technology 176 (2024) 110883.
[37] M. El Marghichi, S. Dangoury, Electrical parameters identification for three diode photovoltaic based on the manta ray foraging optimization with dynamic fitness distance balance, Optik 296 (2023) 171548.
[38] B. Özkaya, Enhanced growth optimizer algorithm with dynamic fitness-distance balance method for solution of security-constrained optimal power flow problem in the presence of stochastic wind and solar energy, Applied Energy 368 (2024) 123499.
[39] S. Aras, E. Gedikli, H. T. Kahraman, A novel stochastic fractal search algorithm with fitness-distance balance for global numerical optimization, Swarm and Evolutionary Computation 61 (2020) 100821.
[40] U. Guvenc, S. Duman, H. T. Kahraman, S. Aras, M. Kati, Fitness-distance balance based adaptive guided differential evolution algorithm for security-constrained optimal power flow problem incorporating renewable energy sources, Applied Soft Computing 108 (04 2021).
[41] H. Bakır, Dynamic fitness-distance balance-based artificial rabbits optimization algorithm to solve optimal power flow problem, Expert Systems with Applications 240 (2024) 122460.
[42] D. Zhu, S. Wang, J. Shen, C. Zhou, T. Li, S. Yan, A multi-strategy particle swarm algorithm with exponential noise and fitness-distance balance method for low-altitude penetration in secure space, Journal of Computational Science 74 (2023) 102149.
[43] Q. Liu, C. Zhang, Z. Li, T. Peng, Z. Zhang, D. Du, M. S. Nazir, Multi-strategy adaptive guidance differential evolution algorithm using fitness-distance balance and opposition-based learning for constrained global optimization of photovoltaic cells and modules, Applied Energy 353 (2024) 122032.
[44] N. Van Thieu, Opfunu: An open-source python library for optimization benchmark functions, Journal of Open Research Software (May 2024).
[45] N. Van Thieu, S. Mirjalili, Mealpy: An open-source library for latest meta-heuristic algorithms in python, Journal of Systems Architecture (2023).
[46] N. Van Thieu, S. D. Barma, T. Van Lam, O. Kisi, A. Mahesha, Groundwater level modeling using augmented artificial ecosystem optimization, Journal of Hydrology 617 (2023) 129034.
[47] A. N. Ahmed, T. Van Lam, N. D. Hung, N. Van Thieu, O. Kisi, A. El-Shafie, A comprehensive comparison of recent developed meta-heuristic algorithms for streamflow time series forecasting problem, Applied Soft Computing 105 (2021) 107282.

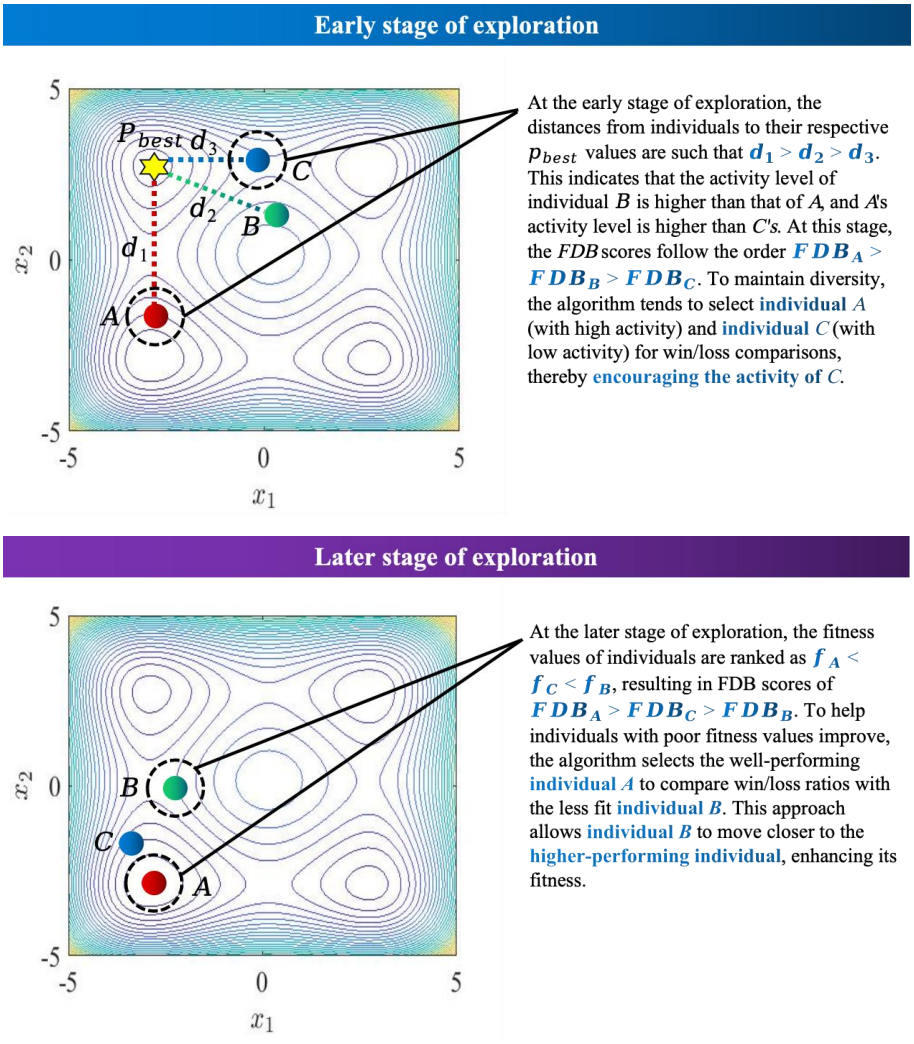


Figure 1: DFDBCSO schematic diagram.

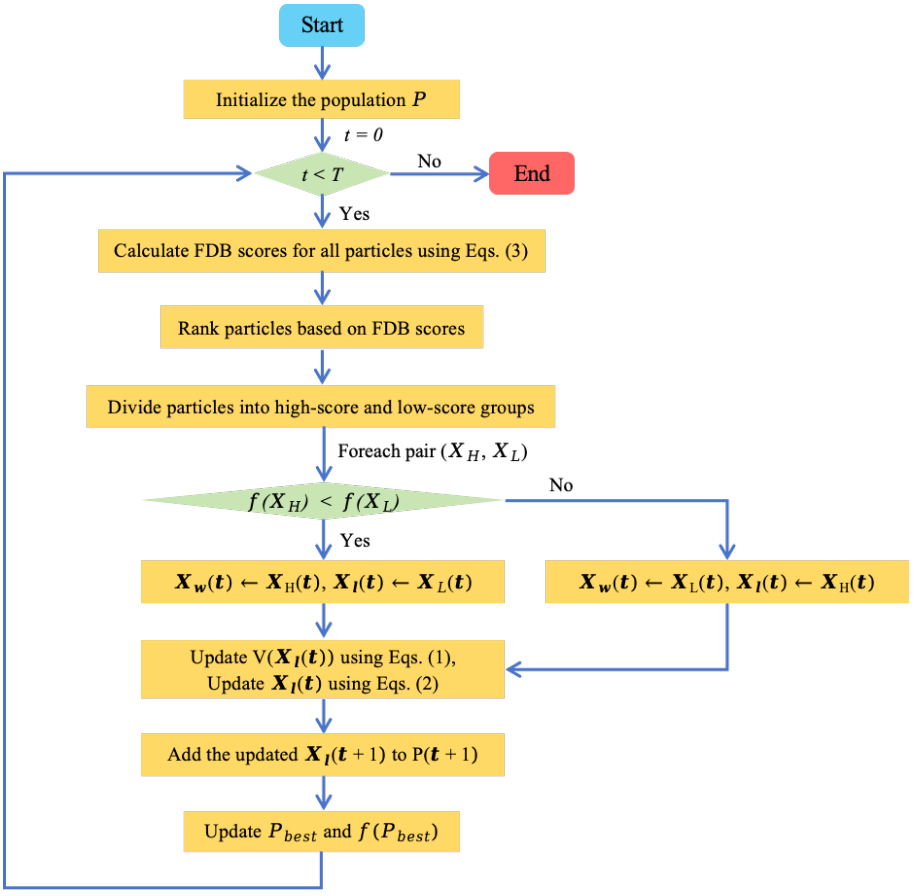


Figure 2: DFDBCSO flow chart.

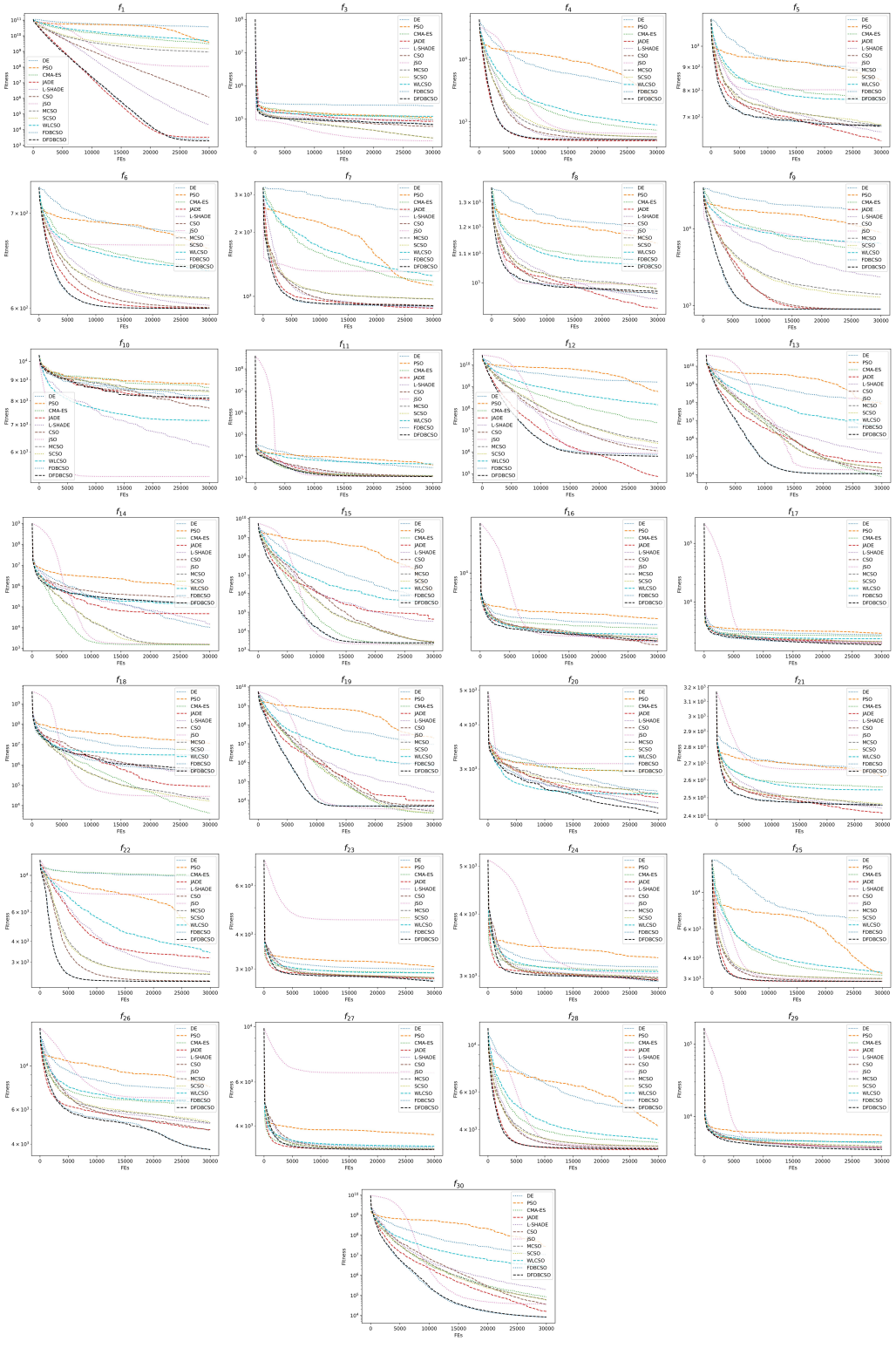


Figure 3: Convergence plot in 30-D CEC2017.

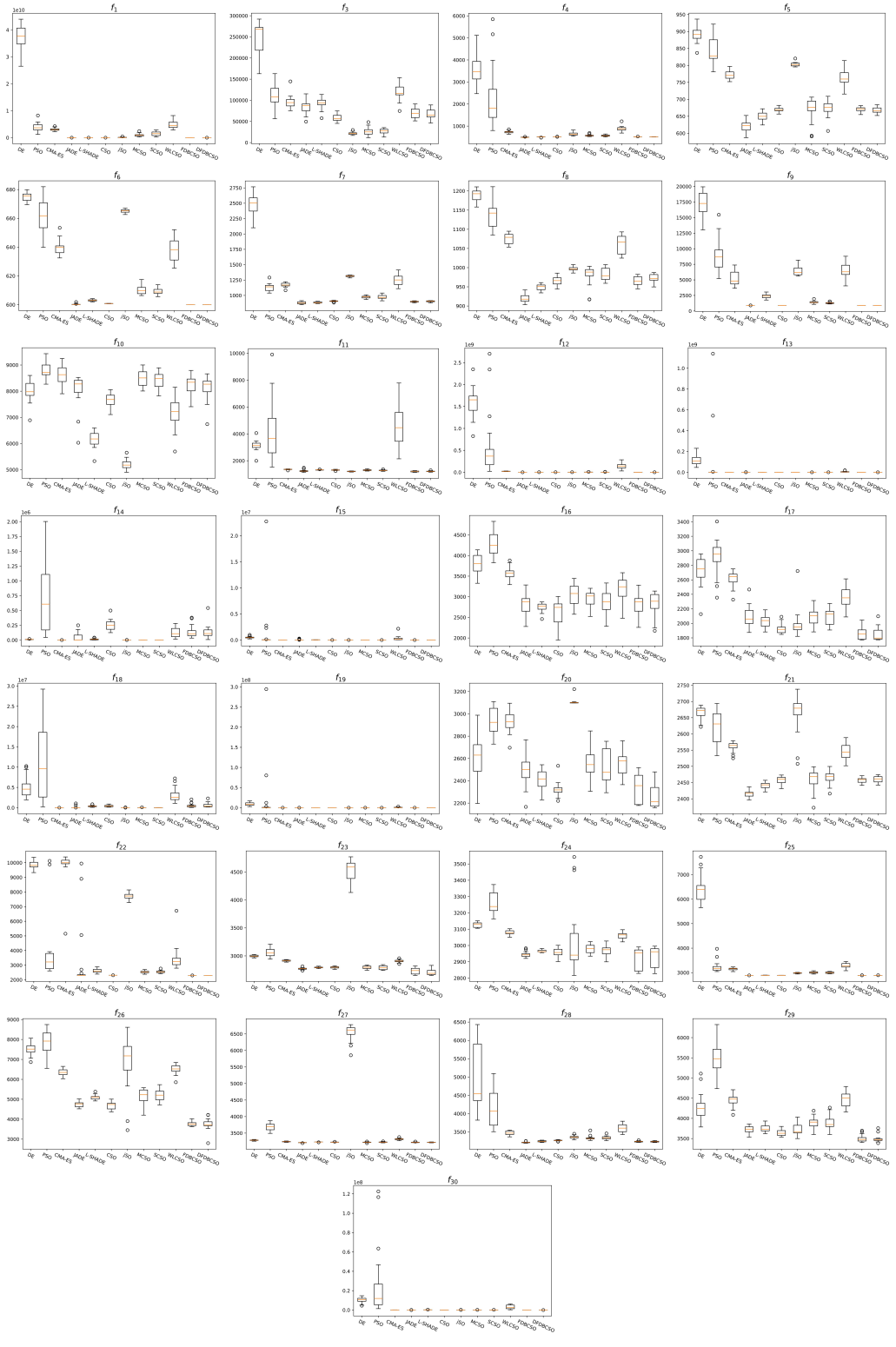


Figure 4: Boxplot plot in 30-D CEC2020.

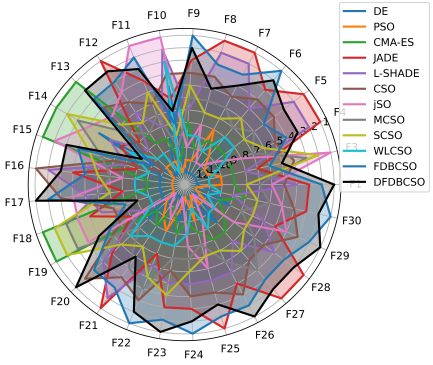


Figure 5: Radar graph of optimizers' ranks in 30-D CEC2017.

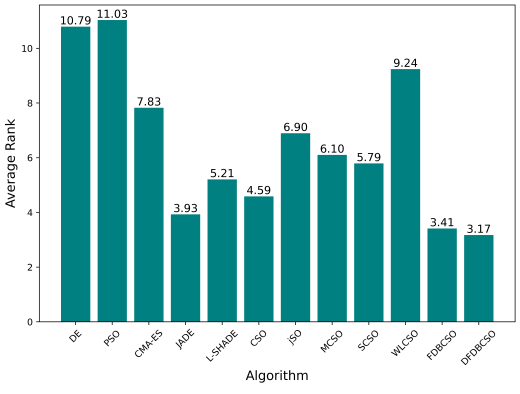


Figure 6: Average ranks of optimizers in 30-D CEC2017.

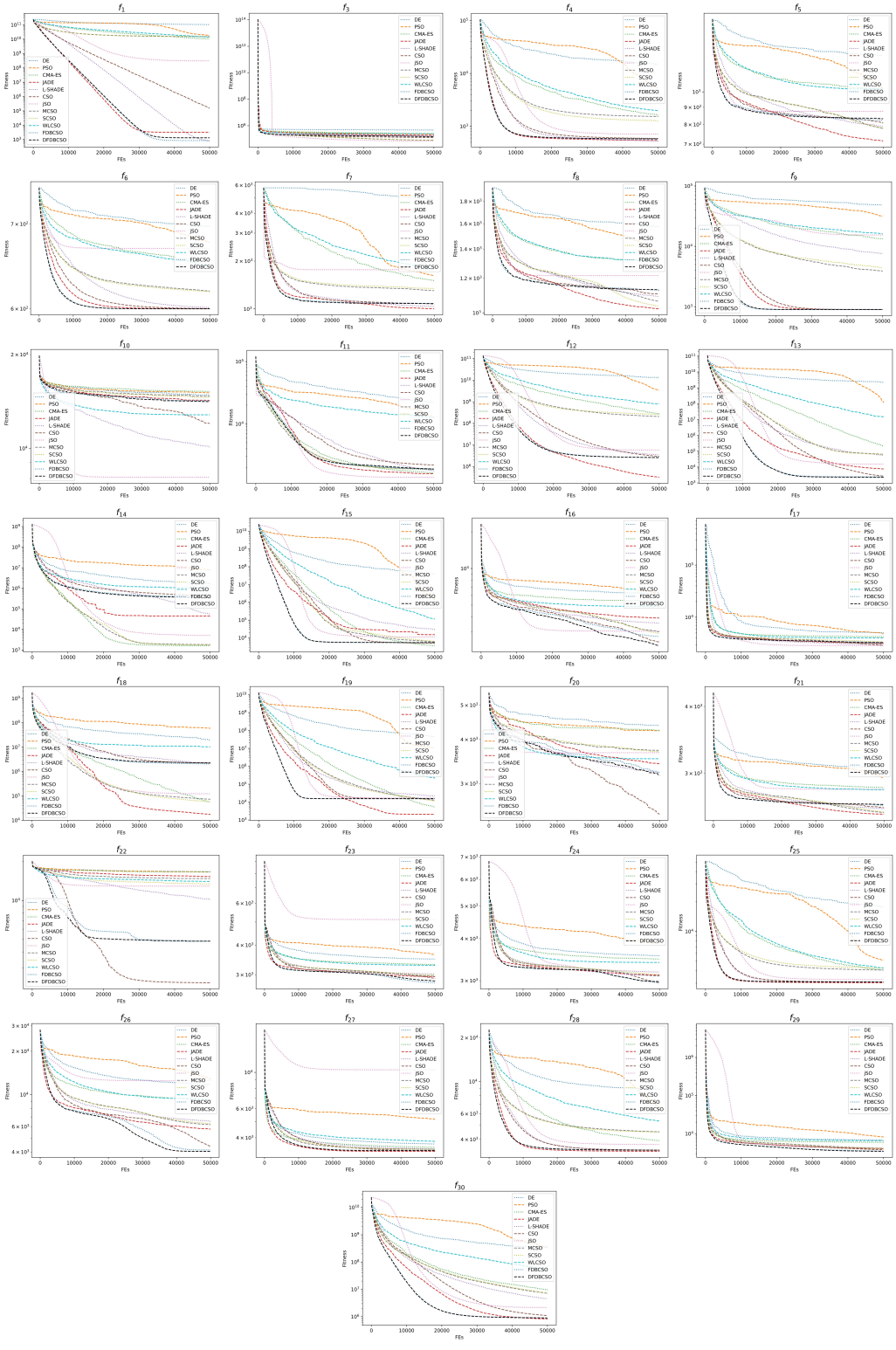


Figure 7: Convergence plot in 50-D CEC2017.

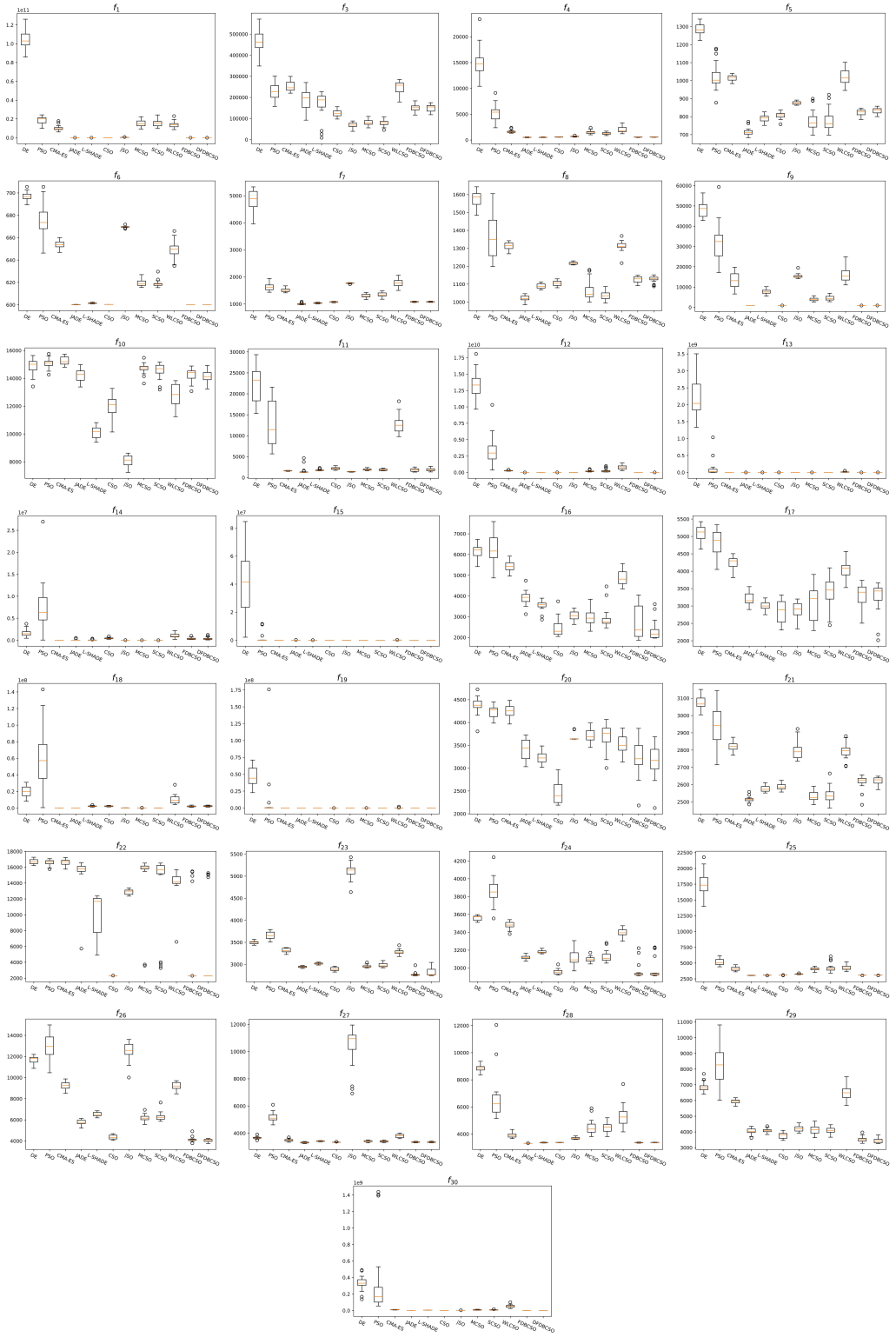


Figure 8: Boxplot plot in 50-D CEC2017.

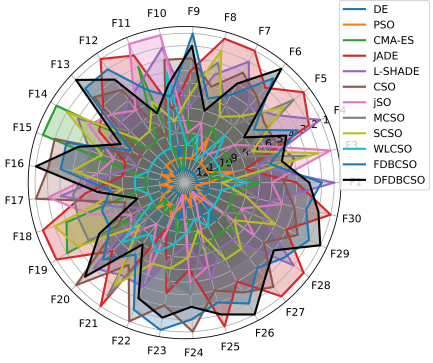


Figure 9: Radar graph of optimizers' ranks in 50-D CEC2017.

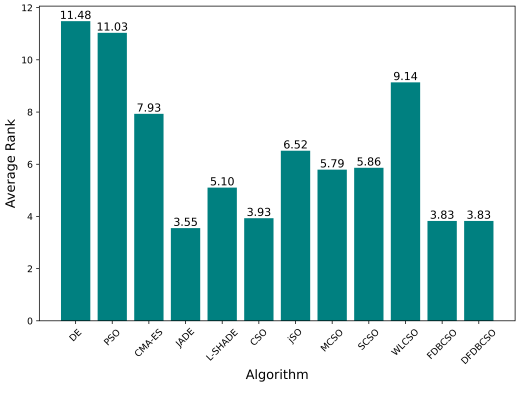


Figure 10: Average ranks of optimizers in 50-D CEC2017.

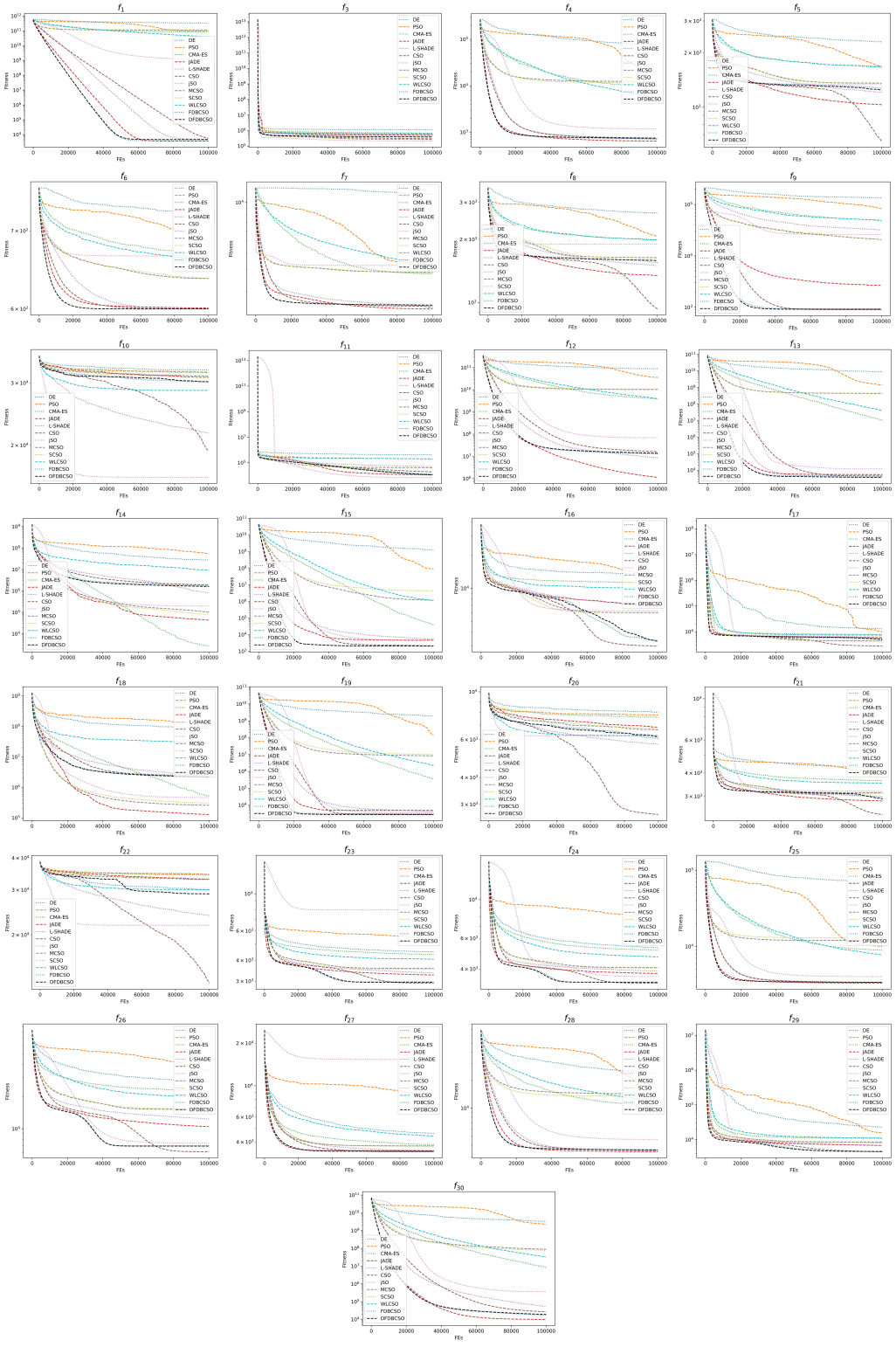


Figure 11: Convergence plot in 100-D CEC2017.

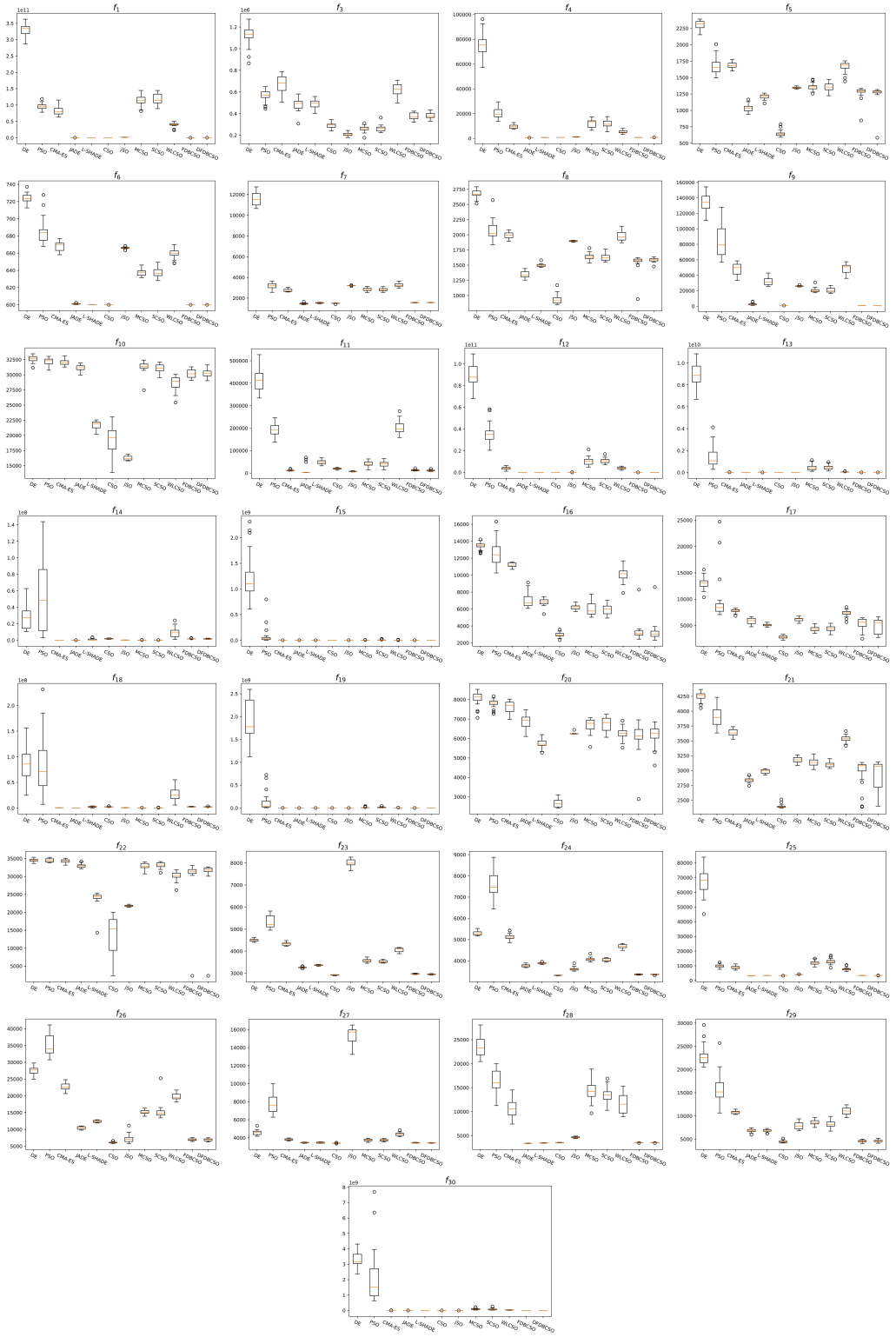


Figure 12: Boxplot plot in 100-D CEC2017.

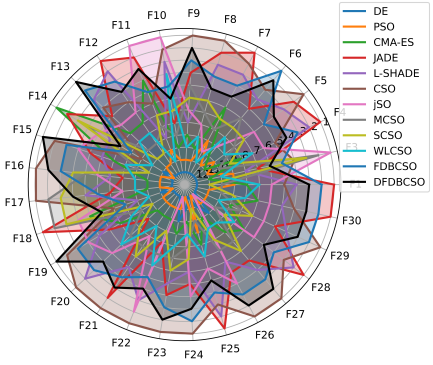


Figure 13: Radar graph of optimizers' ranks in 100-D CEC2017.

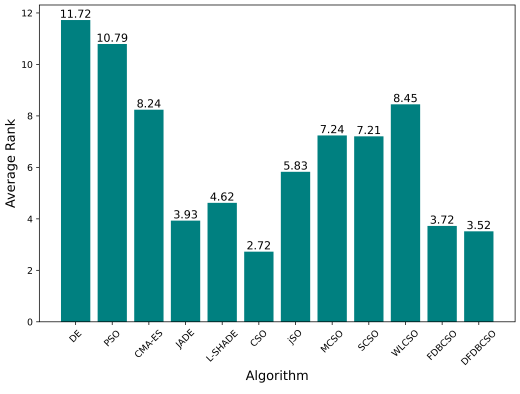


Figure 14: Average ranks of optimizers in 100-D CEC2017.

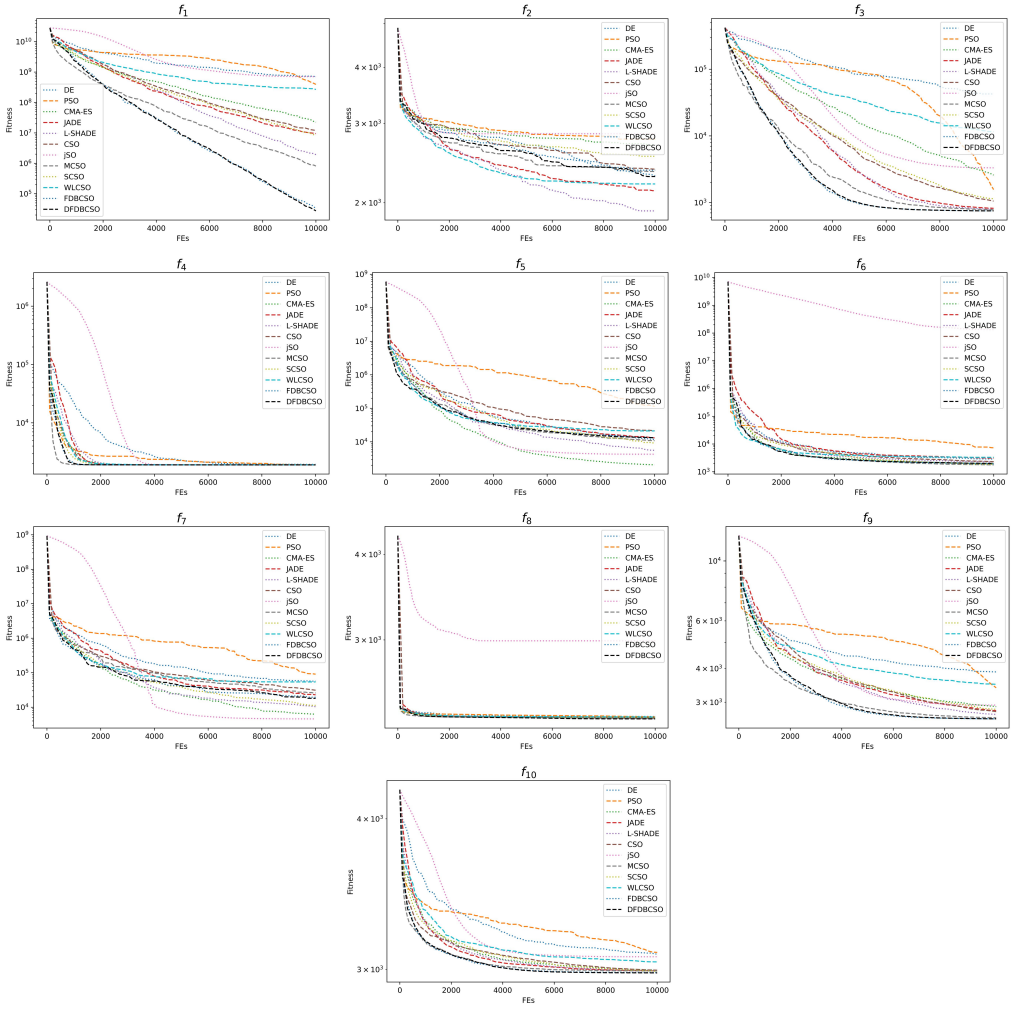


Figure 15: Convergence plot in 10-D CEC2020.

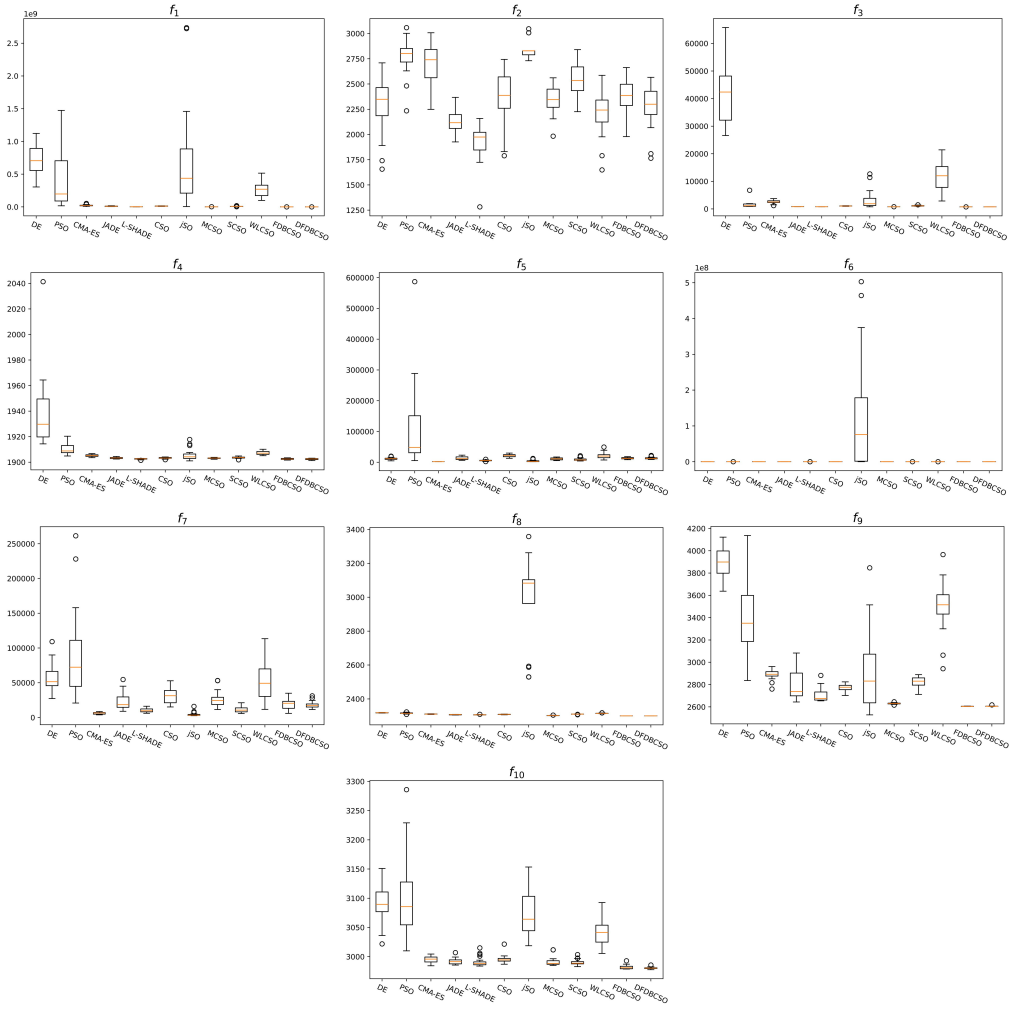


Figure 16: Boxplot plot in 10-D CEC2020.

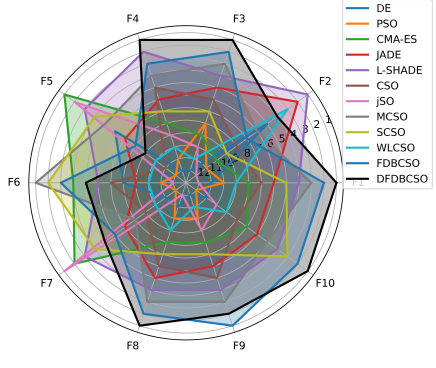


Figure 17: Radar graph of optimizers' ranks in 10-D CEC2020.

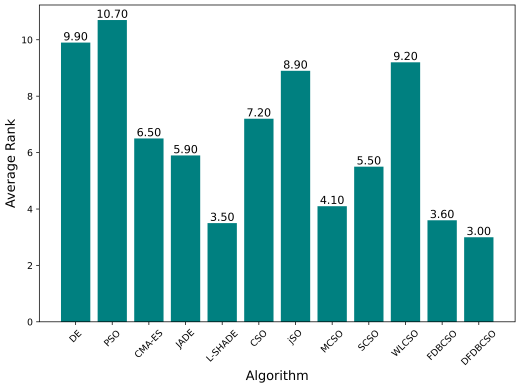


Figure 18: Average ranks of optimizers in 10-D CEC2020.

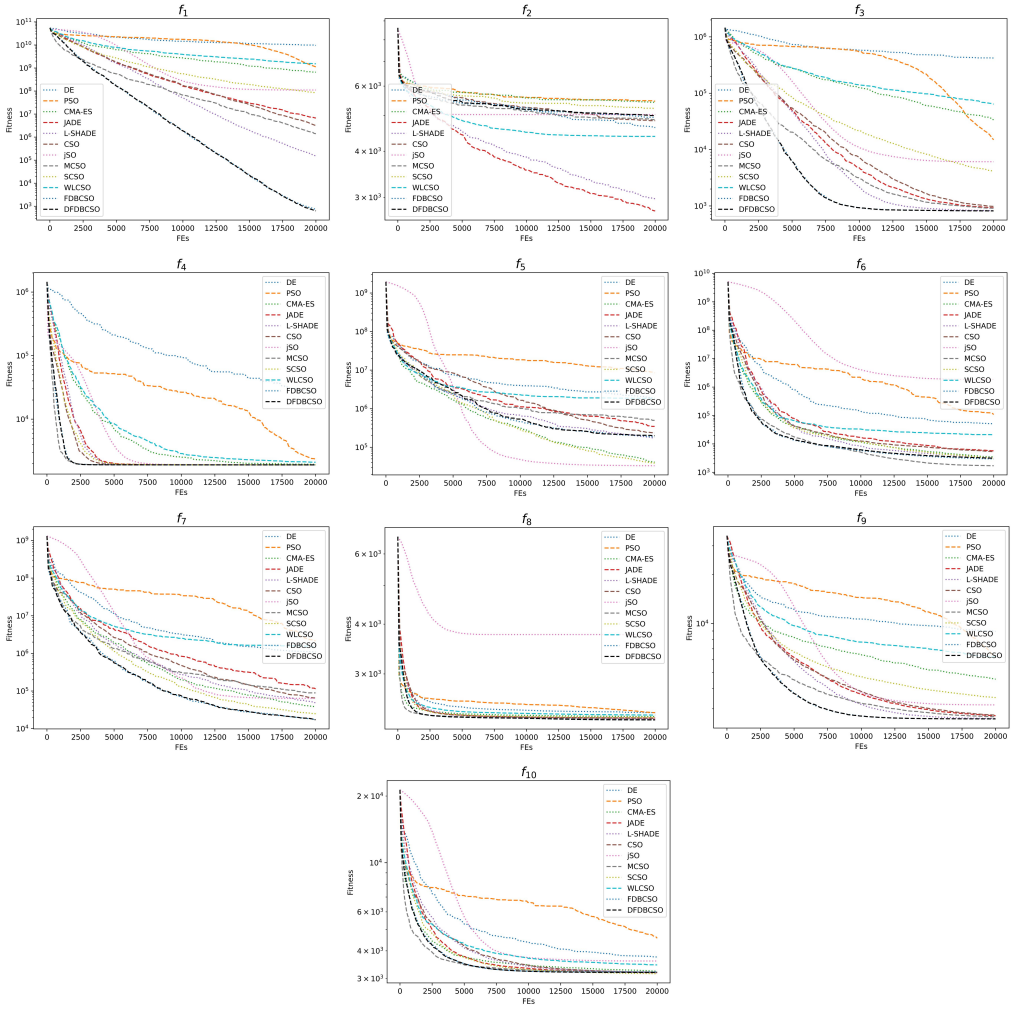


Figure 19: Convergence plot in 20-D CEC2020.

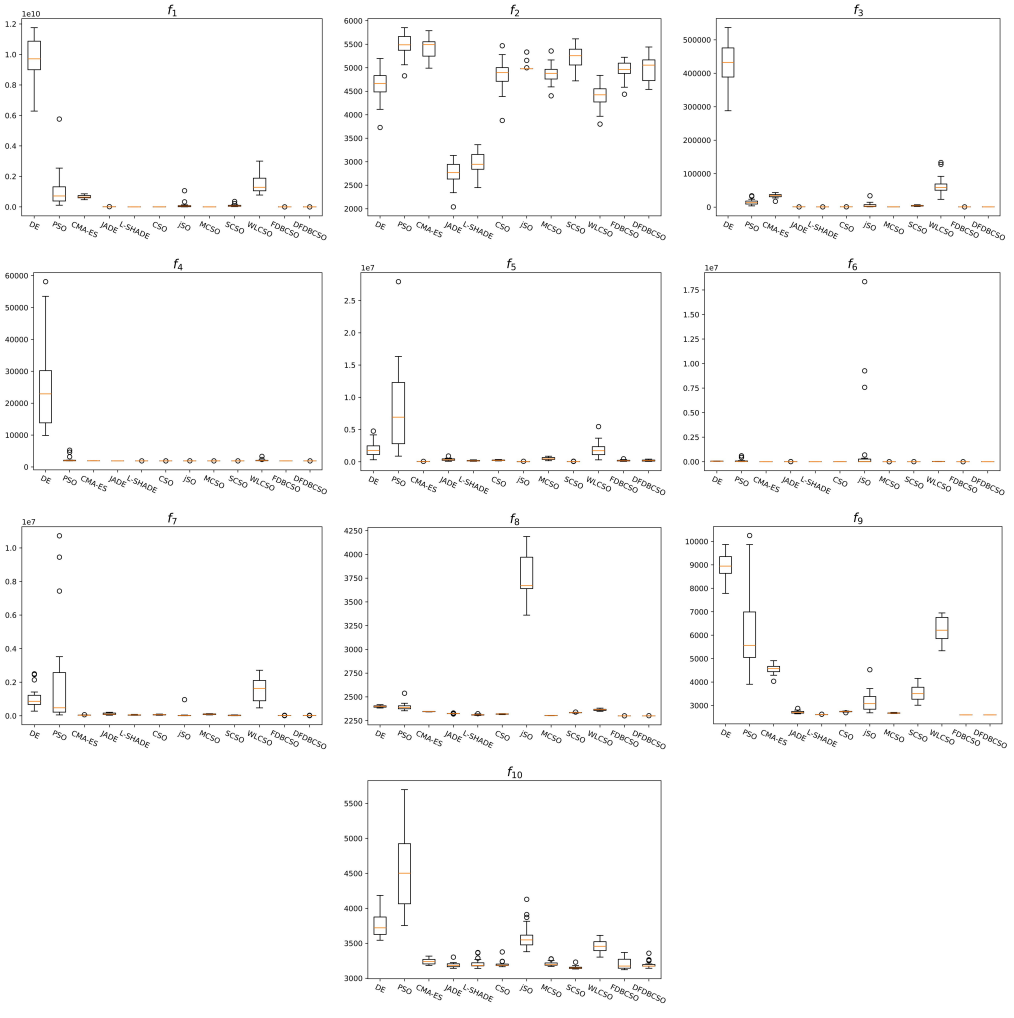


Figure 20: Boxplot plot in 20-D CEC2020.

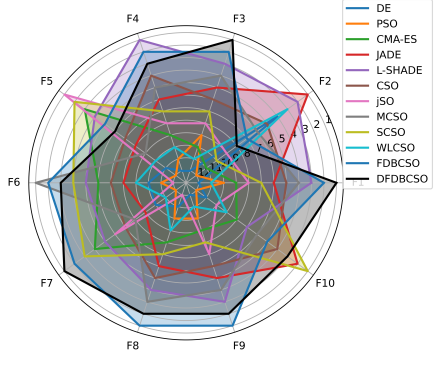


Figure 21: Radar graph of optimizers' ranks in 20-D CEC2020.

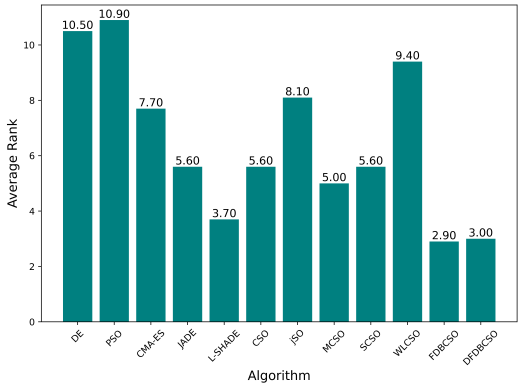


Figure 22: Average ranks of optimizers in 20-D CEC2020.

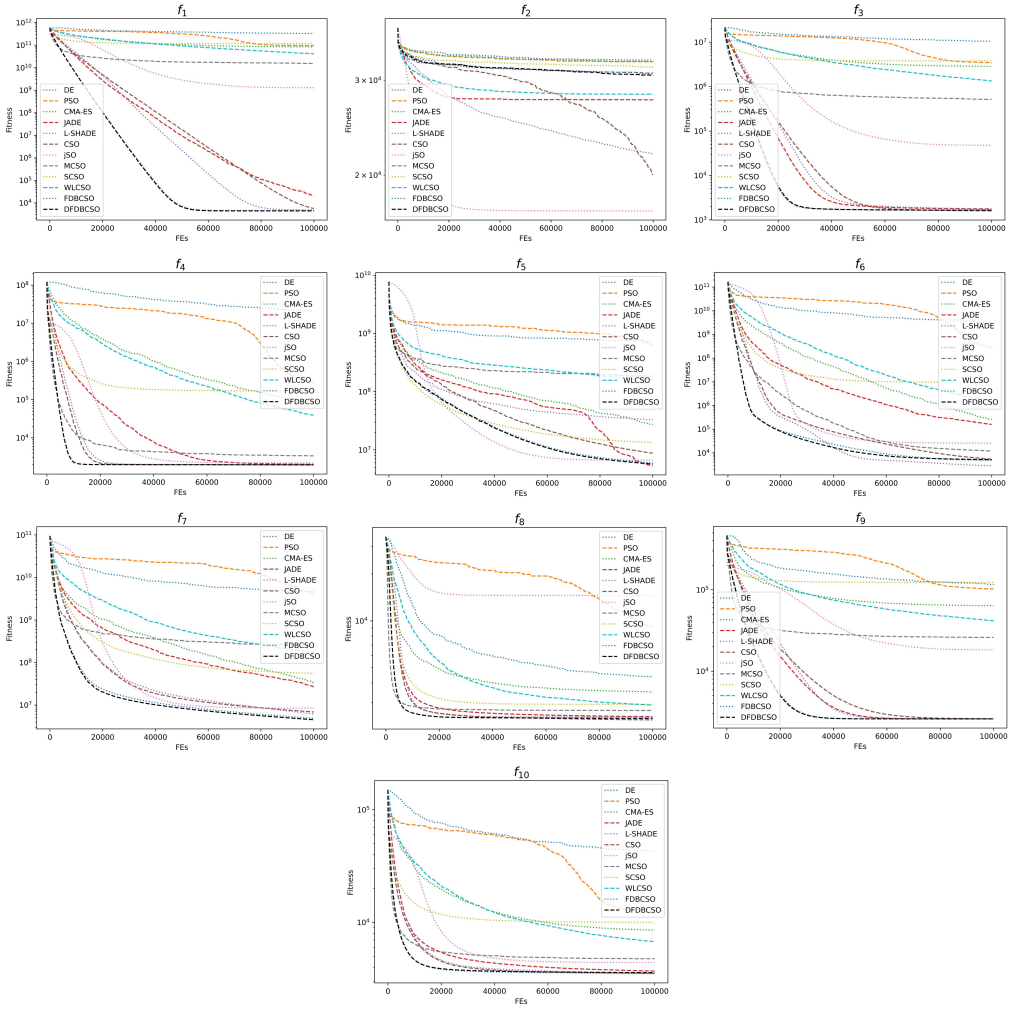


Figure 23: Convergence plot in 100-D CEC2020.

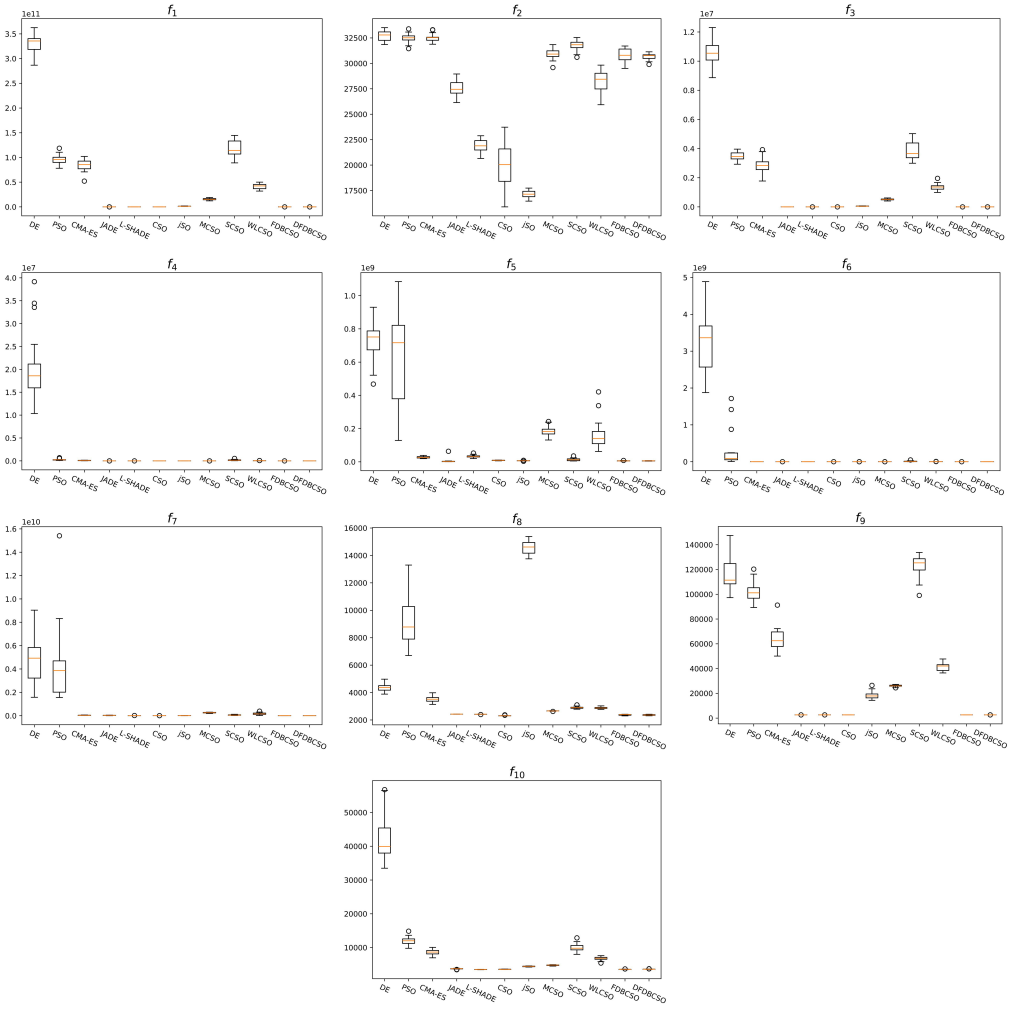


Figure 24: Boxplot plot in 100-D CEC2020.

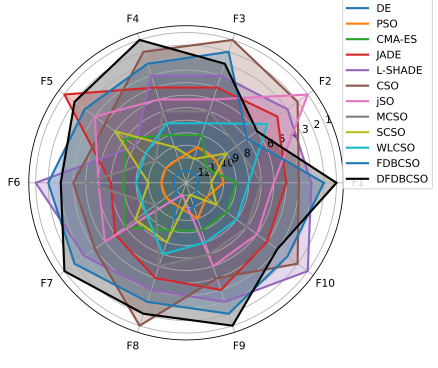


Figure 25: Radar graph of optimizers' ranks in 100-D CEC2020.

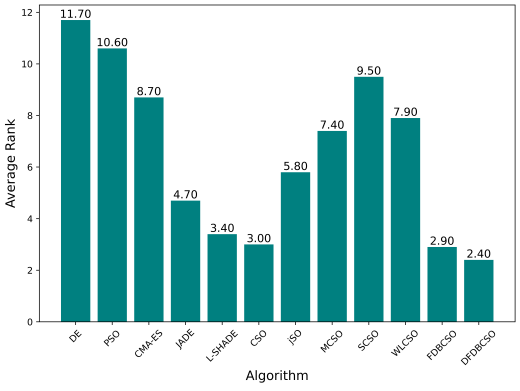


Figure 26: Average ranks of optimizers in 100-D CEC2020.

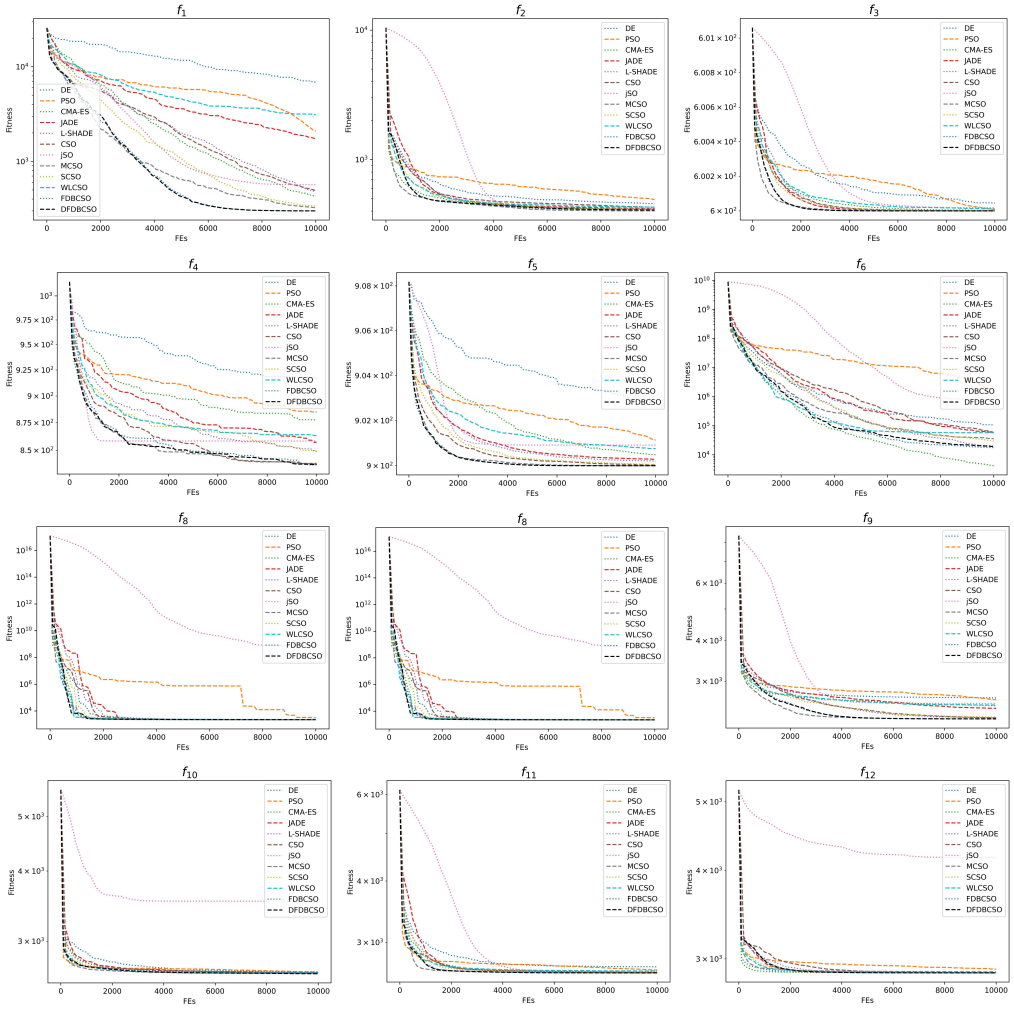


Figure 27: Convergence plot in 10-D CEC2022.

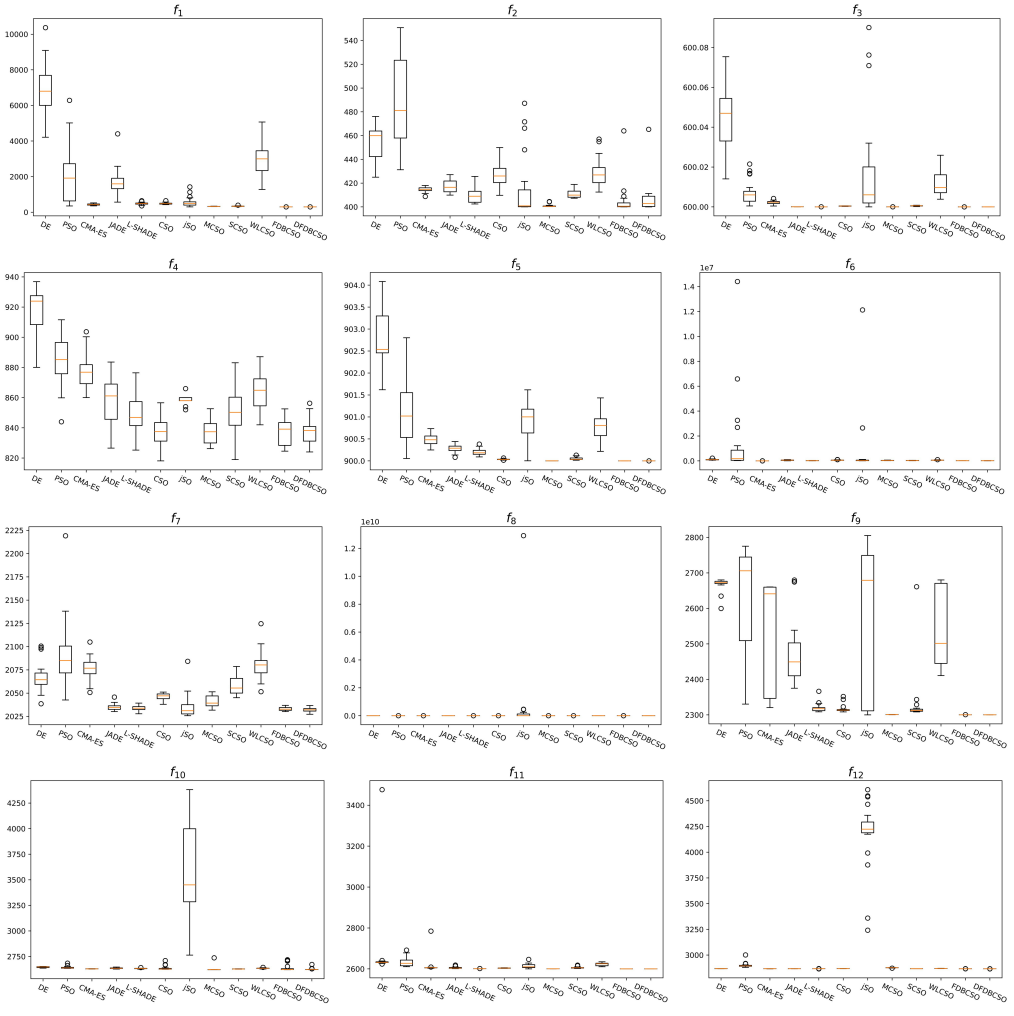


Figure 28: Boxplot plot in 10-D CEC2022.

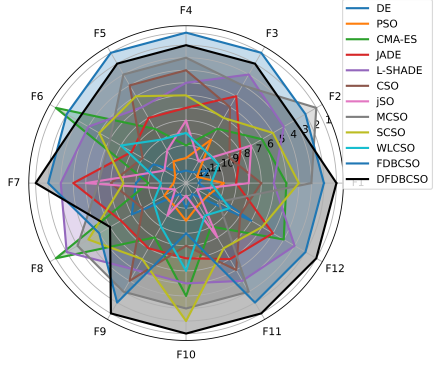


Figure 29: Radar graph of optimizers' ranks in 10-D CEC2022.

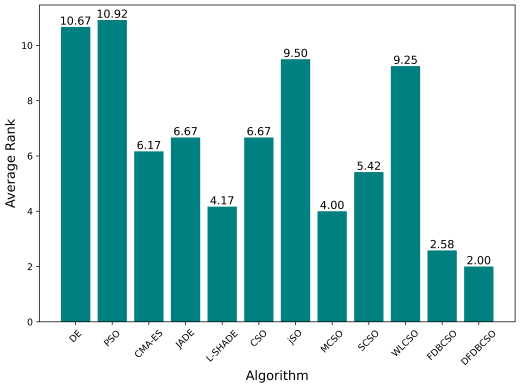


Figure 30: Average ranks of optimizers in 10-D CEC2022.

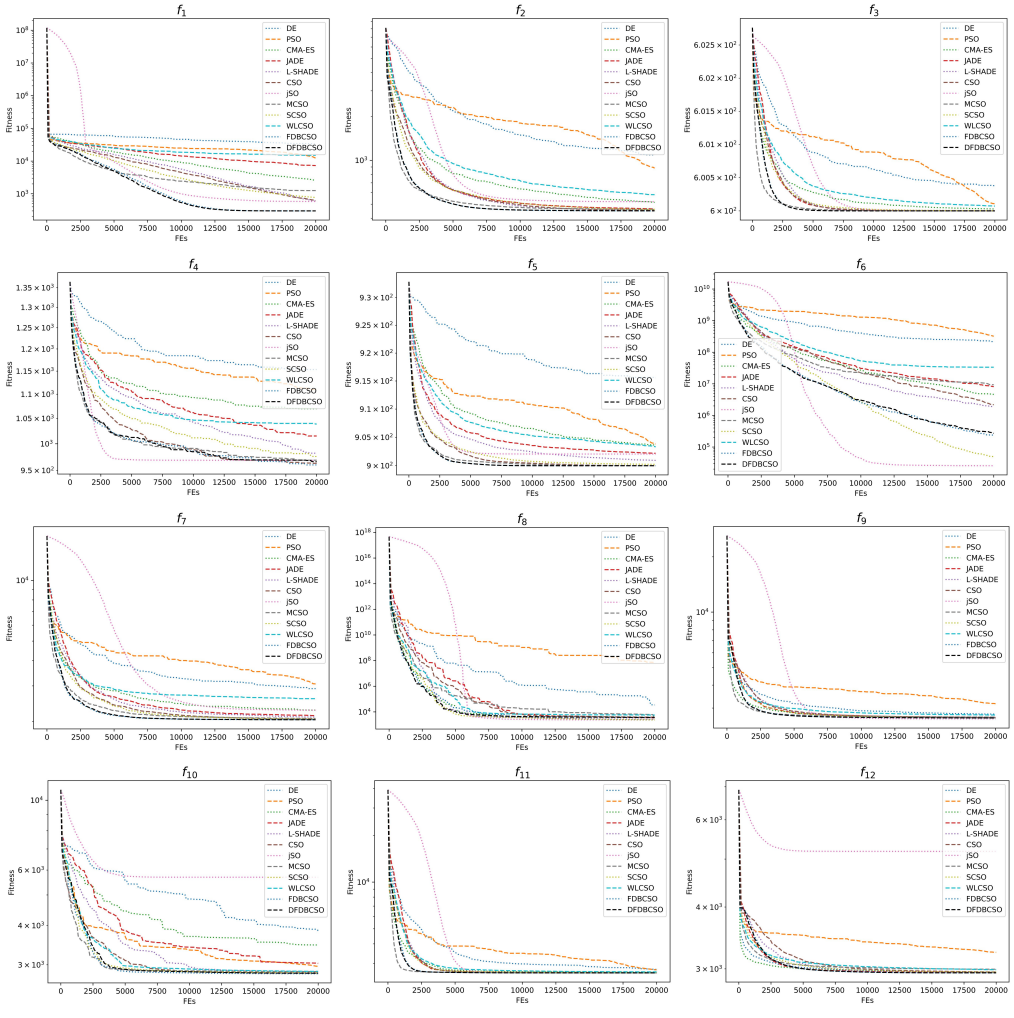


Figure 31: Convergence plot in 20-D CEC2022.

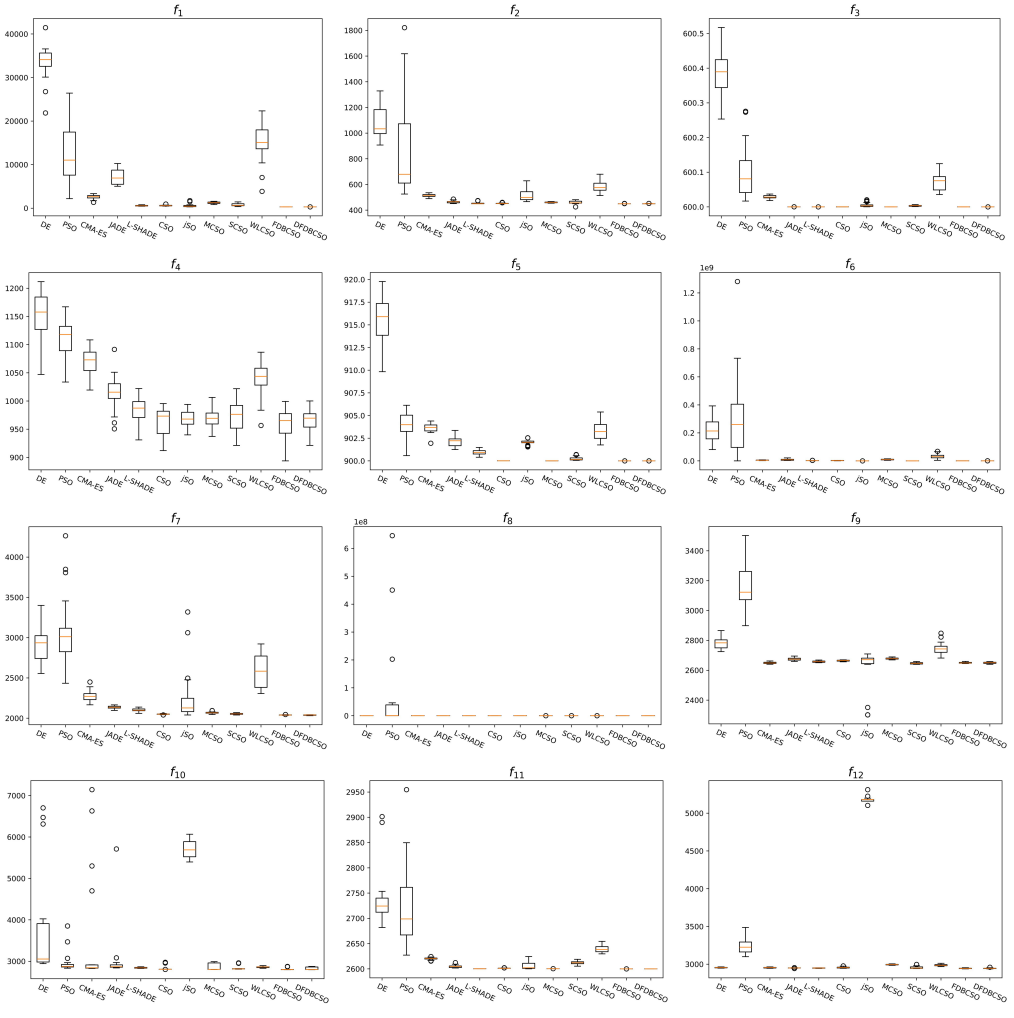


Figure 32: Boxplot plot in 20-D CEC2022.

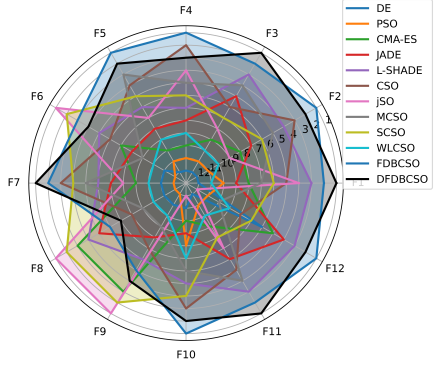


Figure 33: Radar graph of optimizers' ranks in 20-D CEC2022.

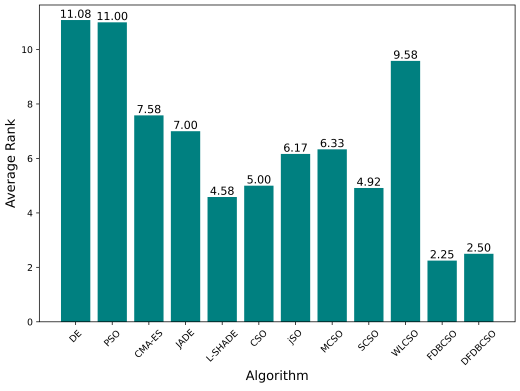


Figure 34: Average ranks of optimizers in 20-D CEC2022.

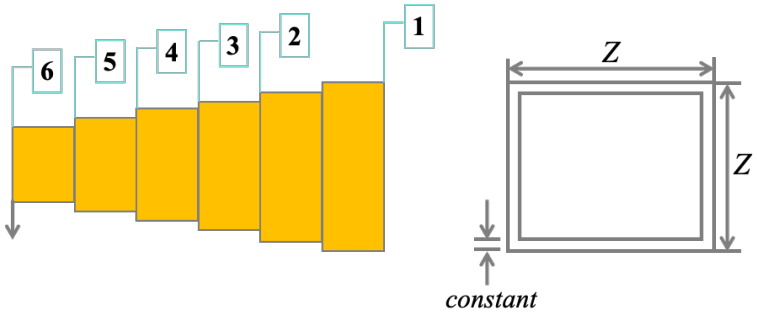


Figure 35: Illustrative diagram of the cantilever beam design problem.

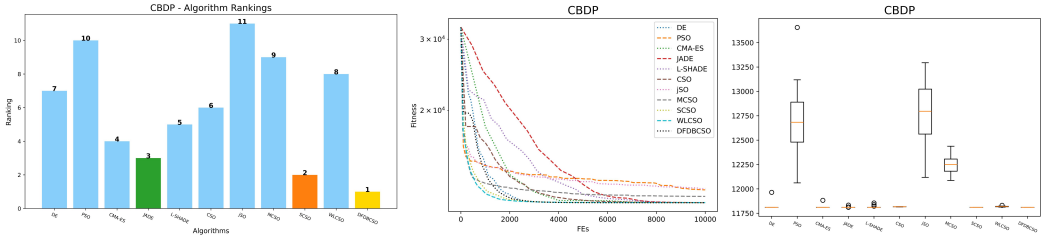


Figure 36: Optimizer rankings, convergence trends, and boxplot analyses for CBDP.

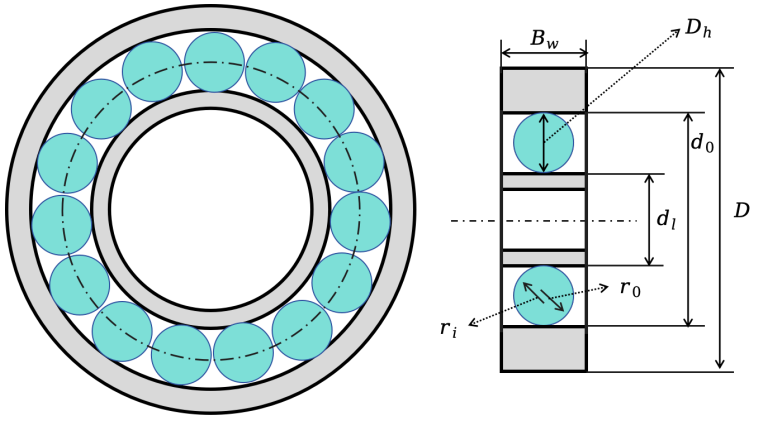


Figure 37: Illustrative diagram of the rolling element bearing design problem.

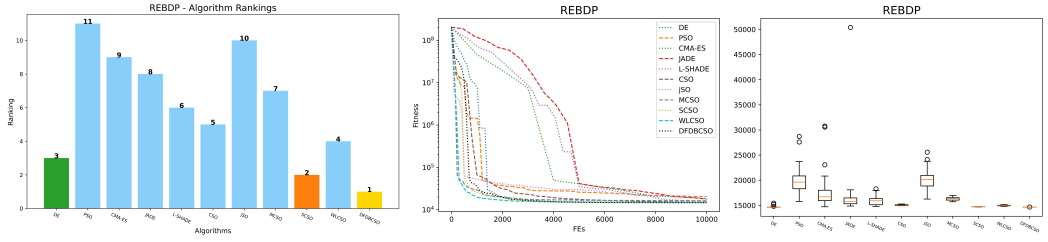


Figure 38: Optimizer rankings, convergence trends, and boxplot analyses for REBDP.

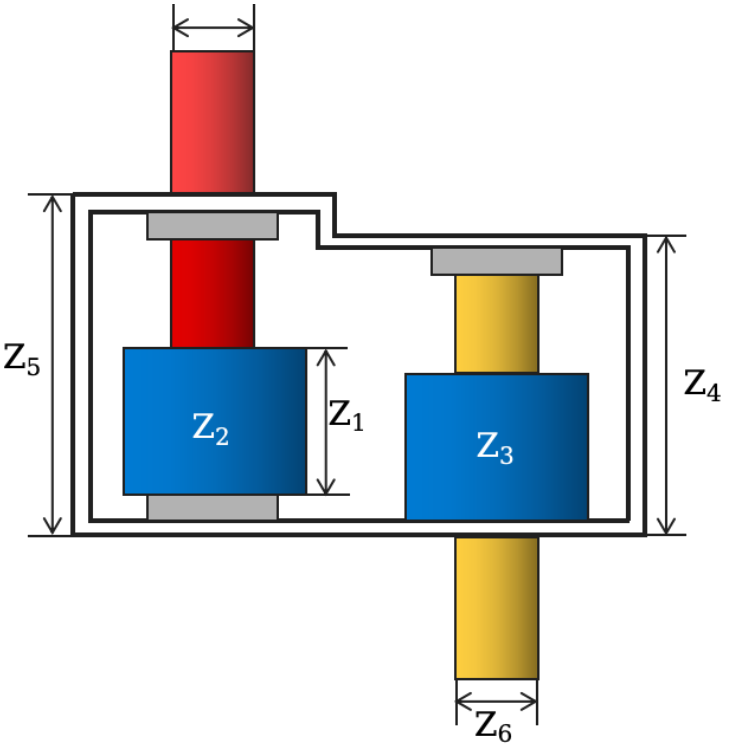


Figure 39: Illustrative diagram of the speed reducer design problem.

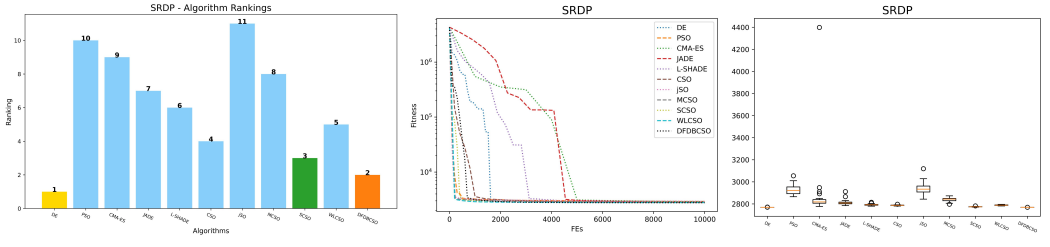


Figure 40: Optimizer rankings, convergence trends, and boxplot analyses for SRDP.



Figure 41: Illustrative diagram of the tension/compression spring design problem.

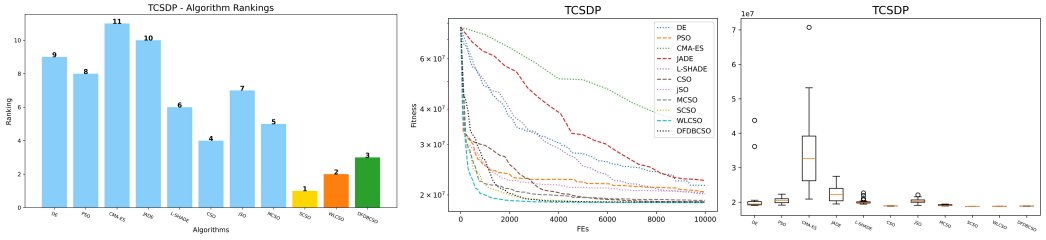


Figure 42: Optimizer rankings, convergence trends, and boxplot analyses for TCSDP.

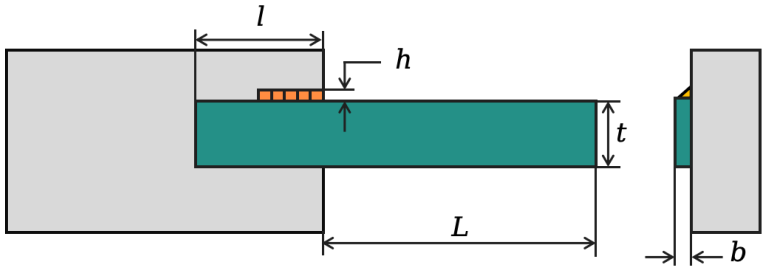


Figure 43: Illustrative diagram of the welded beam design problem.

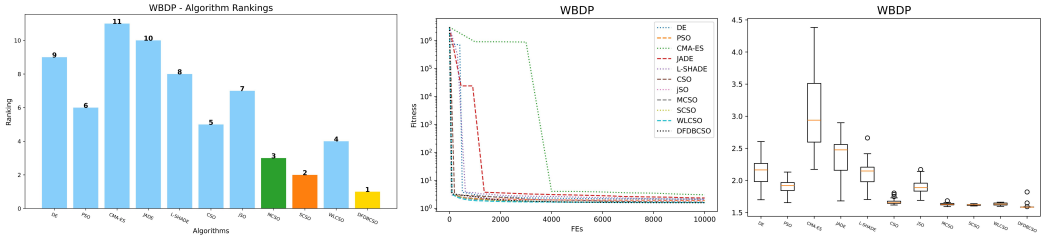


Figure 44: Optimizer rankings, convergence trends, and boxplot analyses for WBDP.

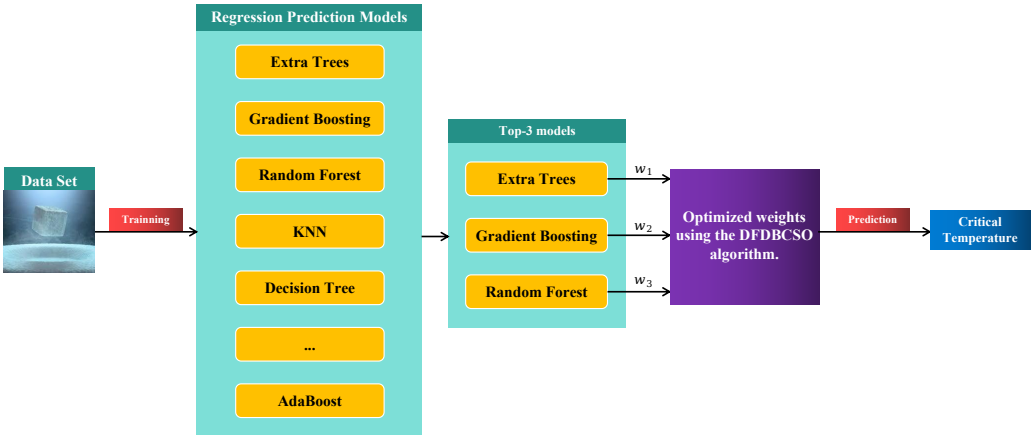


Figure 45: The architecture of DFDBCSO-Ensemble.

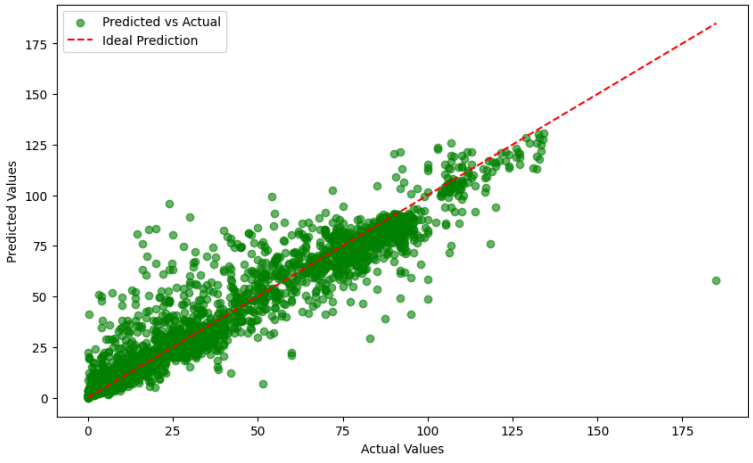


Figure 46: Actual vs. Predicted plot of DFDBCSO-Ensemble.

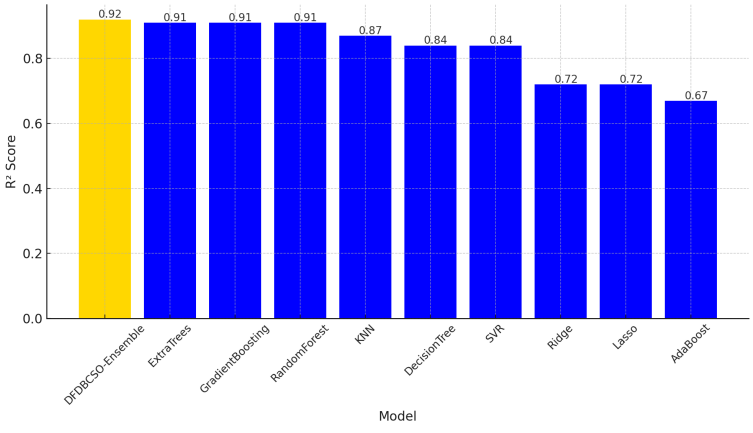


Figure 47: Model performance in superconductor critical temperature prediction.

A. Detailed Results in CEC2017

Tables 11, 12 and 13 provide a comprehensive summary of the experimental results for the optimizers on the CEC2017 benchmark.

B. Detailed Results in CEC2020

Tables 14, 15 and 16 provide a comprehensive summary of the experimental results for the optimizers on the CEC2020 benchmark.

C. Detailed Results in CEC2022

Tables 17 and 18 provide a comprehensive summary of the experimental results for the optimizers on the CEC2022 benchmark.

D. Mathematical Model of Engineering Problems

This section introduces the mathematical formulation of engineering problems.

D.1. Mathematical Model of Cantilever Beam Design Problem (CBDP)

The Mathematical Model of the Cantilever Beam Design Problem (CBDP) is expressed in Eq. (4).

$$\begin{aligned}
&\textbf{Objective Function:} Minimize f(\mathbf{z}) = 0.6224(z_1 + z_2 + z_3 + z_4 + z_5) \\
&\textbf{Constraint:} g(\mathbf{z}) = \frac{61}{z_1^3} + \frac{27}{z_2^3} + \frac{19}{z_3^3} + \frac{7}{z_4^3} + \frac{1}{z_5^3} - 1 \leq 0 \\
&\textbf{Variable Range:} 0.01 \leq z_1, z_2, z_3, z_4, z_5 \leq 100
\end{aligned} \tag{4}$$

D.2. Mathematical Model of Rolling Element Bearing Design Problem

The Rolling Element Bearing Design Problem (REBDP) is mathematically formulated in Eqs. (5), (6), and (7), aiming to maximize the dynamic load-carrying capacity (C_d). This involves ten geometric parameters: pitch diameter (D_m), ball diameter (D_b), number of balls (Z), curvature radii coefficients (f_i, f_o), radial clearance limits ($K_{\text{dmin}}, K_{\text{dmax}}$), load-deflection ratio (δ), and other related parameters (e, ζ).

$$C_d = \begin{cases} f_c Z^{2/3} D_b^{1.8}, & \text{if } D_b \leq 25.4\,\text{mm}, \\ 3.647 f_c Z^{2/3} D_b^{1.4}, & \text{otherwise.} \end{cases} \tag{5}$$

Subject to:

$$\begin{aligned}
g_1(\mathbf{z}) &= \frac{\phi_0}{2\sin^{-1}(D_b/D_m)} - Z + 1 \leq 0, \\
g_2(\mathbf{z}) &= 2D_b - K_{\text{dmin}}(D - d) > 0, \\
g_3(\mathbf{z}) &= K_{\text{dmax}}(D - d) - 2D_b \geq 0, \\
g_4(\mathbf{z}) &= \zeta B_w - D_b \leq 0, \\
g_5(\mathbf{z}) &= D_m - 0.5(D + d) \geq 0, \\
g_6(\mathbf{z}) &= (0.5 + e)(D + d) - D_m \geq 0, \\
g_7(\mathbf{z}) &= 0.5(D - D_m - D_b) - \delta D_b \geq 0, \\
g_8(\mathbf{z}) &= f_i \geq 0.515, \\
g_9(\mathbf{z}) &= f_o \geq 0.515.
\end{aligned} \tag{6}$$

Where:

$$
\begin{aligned}
f_c &= 37.91\left[1+\left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\left(\frac{f_i(2f_o-1)}{f_o(2f_i-1)}\right)^{0.41}\right\}^{\frac{10}{3}}\right]^{-0.3} \\
&\quad\times\left[\frac{\gamma^{0.3}(1-\gamma)^{1.39}}{(1+\gamma)^{\frac{1}{3}}}\right]\left[\frac{2f_i}{2f_i-1}\right]^{0.41}, \\
x &= \left\{\frac{D-d}{2}-3\left(\frac{T}{4}\right)\right\}^2+\left\{\frac{D}{2}-\frac{T}{4}-D_b\right\}^2-\left\{\frac{d}{2}+\frac{T}{4}\right\}^2, \\
y &= 2\left\{\frac{D-d}{2}-3\left(\frac{T}{4}\right)\right\}\left\{\frac{D}{2}-\frac{T}{4}-D_b\right\}, \\
\phi_0 &= 2\pi-\cos^{-1}\left(\frac{x}{y}\right), \\
\gamma &= \frac{D_b}{D_m}, \quad f_i=\frac{r_i}{D_b}, \quad f_o=\frac{r_o}{D_b}, \\
T &= D-d-2D_b, \quad D=160, \quad d=90, \quad B_w=30, \quad r_i=r_o=11.033, \\
0.5(D+d) &\le D_m \le 0.6(D+d), \quad 0.15(D-d)\le D_b\le 0.45(D-d), \\
4 &\le Z\le 50, \quad 0.515\le f_i, f_o\le 0.6, \quad 0.4\le K_{\text{dmin}}\le 0.5, \\
0.6 &\le K_{\text{dmax}}\le 0.7, \quad 0.3\le\delta\le 0.4, \quad 0.02\le e\le 0.1, \quad 0.6\le\zeta\le 0.85.
\end{aligned}
\tag{7}
$$

D.3. Mathematical Model of Speed Reducer Design Problem

The Mathematical Model of the Speed Reducer Design Problem (SRDP) is expressed in Eq. (8), (9), (10).

Objective Function

$$
f(\mathbf{z}) = 0.7854z_1z_2^2\left(3.3333z_3^2+14.9334z_3-43.0934\right)-1.508z_1\left(z_6^2+z_7^2\right)+7.4777\left(z_6^3+z_7^3\right) \tag{8}
$$

Constraints

$$
\begin{aligned}
g_1(\mathbf{z}) &= \frac{27}{z_1z_2^2z_3}-1\le 0, \quad g_2(\mathbf{z})=\frac{397.5}{z_1z_2^2z_3^2}-1\le 0, \\
g_3(\mathbf{z}) &= \frac{1.93z_4^3}{z_2z_3z_6^4}-1\le 0, \quad g_4(\mathbf{z})=\frac{1.93z_5^3}{z_2z_3z_7^4}-1\le 0, \\
g_5(\mathbf{z}) &= \frac{\sqrt{\left(\frac{745z_4}{z_2z_3}\right)^2+1.69\times 10^6}}{110z_6^3}-1\le 0, \\
g_6(\mathbf{z}) &= \frac{\sqrt{\left(\frac{745z_4}{z_2z_3}\right)^2+157.5\times 10^6}}{85z_6^3}-1\le 0, \\
g_7(\mathbf{z}) &= \frac{z_2z_3}{40}-1\le 0, \quad g_8(\mathbf{z})=\frac{5z_2}{z_1}-1\le 0, \\
g_9(\mathbf{z}) &= \frac{z_1}{12z_2}-1\le 0, \quad g_{10}(\mathbf{z})=\frac{1.5z_6+1.9}{z_4}-1\le 0, \\
g_{11}(\mathbf{z}) &= \frac{1.1z_7+1.9}{z_5}-1\le 0.
\end{aligned}
\tag{9}
$$

Variable Bounds

$$
\begin{aligned}
&2.6\le z_1\le 3.6, \quad 0.7\le z_2\le 0.8, \quad 17\le z_3\le 28, \\
&7.3\le z_4\le 8.3, \quad 7.8\le z_5\le 8.3, \quad 2.9\le z_6\le 3.9, \quad 5.0\le z_7\le 5.5.
\end{aligned}
\tag{10}
$$

D.4. Mathematical Model of Tension/compression Spring Design Problem

The Mathematical Model of the Tension/compression Spring Design Problem (TCSDP) is expressed in Eq. (11), (12), (13).

Objective Function

$$f(\mathbf{z}) = (z_3 + 2) z_2 z_1^2 \tag{11}$$

Constraints

$$\begin{aligned} g_1(\mathbf{z}) &= 1 - \frac{z_2^3 z_3}{71785 z_1^4} \leq 0, \\ g_2(\mathbf{z}) &= \frac{4z_2^2 - z_1 z_2}{12566(z_2 z_1^3 - z_1^4)} + \frac{1}{5108 z_1^2} - 1 \leq 0, \\ g_3(\mathbf{z}) &= 1 - \frac{140.45 z_1}{z_2^2 z_3} \leq 0, \\ g_4(\mathbf{z}) &= \frac{z_1 + z_2}{1.5} - 1 \leq 0. \end{aligned} \tag{12}$$

Variable Bounds

$$0.05 \leq z_1 \leq 2.00, \quad 0.25 \leq z_2 \leq 1.30, \quad 2.00 \leq z_3 \leq 15.00. \tag{13}$$

D.5. Mathematical Model of Welded Beam Design Problem

The Mathematical Model of the Model of Welded Beam Design Problem (WBDP) is expressed in Eq. (14), (15), (16), (17), (18).

Objective Function

$$f(\mathbf{z}) = 1.10471 z_1^2 z_2 + 0.04811 z_3 z_4 (14 + z_2) \tag{14}$$

Constraints Objective Function

$$\begin{aligned} g_1(\mathbf{z}) &= \tau(z) - \tau_{\max} \leq 0, \\ g_2(\mathbf{z}) &= \sigma(z) - \sigma_{\max} \leq 0, \\ g_3(\mathbf{z}) &= \delta(z) - \delta_{\max} \leq 0, \\ g_4(\mathbf{z}) &= z_1 - z_4 \leq 0, \\ g_5(\mathbf{z}) &= P - P_c(z) \leq 0, \\ g_6(\mathbf{z}) &= 0.125 - z_1 \leq 0, \\ g_7(\mathbf{z}) &= 1.10471 z_1^2 + 0.04811 z_3 z_4 \left(14 + z_2\right) - 5 \leq 0. \end{aligned} \tag{15}$$

Variable Bounds

$$0.1 \leq z_1, z_4 \leq 2, 0.1 \leq z_2, z_3 \leq 10 \tag{16}$$

Definitions

$$
\begin{aligned}
\tau(z) &= \sqrt{(\tau')^2 + 2\tau'\tau''\frac{z_2}{2R} + (\tau'')^2}, \\
\tau' &= \frac{P}{\sqrt{2}z_1 z_2}, \\
\tau'' &= \frac{MR}{J}, \\
M &= P\left(L + \frac{z_2}{2}\right), \\
R &= \sqrt{\frac{z_2^2}{4} + \left(\frac{z_1 + z_3}{2}\right)^2}, \\
J &= 2\left\{\sqrt{2}z_1 z_2\left[\frac{z_2^2}{4} + \left(\frac{z_1 + z_3}{2}\right)^2\right]\right\}, \\
\sigma(z) &= \frac{6PL}{Ez_3^2 z_4}, \\
\delta(z) &= \frac{6PL^3}{Ez_3^2 z_4}, \\
P_C(z) &= \frac{4.013E\sqrt{\frac{z_3^2 z_4^6}{36}}}{L^2}\left(1 - \frac{z_3}{2L}\sqrt{\frac{E}{4G}}\right), \\
P &= 6000\,\text{lb}.
\end{aligned}
\tag{17}
$$

Constants

$$
\begin{aligned}
L &= 14\,\text{in}, \\
E &= 30 \times 10^6\,\text{psi}, \\
G &= 12 \times 10^6\,\text{psi}, \\
\delta_{\max} &= 0.25\,\text{in}, \\
\tau_{\max} &= 13600\,\text{psi}, \\
\sigma_{\max} &= 30000\,\text{psi}.
\end{aligned}
\tag{18}
$$

Table 11
Detailed experimental results on 30-D CEC2017.

Func.		DE	PSO	CMA-ES	JADE	L-SHADE	CSO	jSO	MCSO	SCSO	WLCSO	FDBCSO	DFDBCSO
f_1	mean	3.729e+10 +	3.925e+09 +	3.089e+09 +	3.352e+03 ≈	2.117e+04 +	1.208e+06 +	1.111e+08 +	9.503e+08 +	1.497e+09 +	4.929e+09 +	2.472e+03 +	**1.973e+03**
	std	4.261e+09	1.651e+09	5.223e+08	4.567e+03	1.575e+04	1.897e+05	1.096e+08	5.073e+08	7.679e+08	1.472e+09	1.705e+03	1.514e+03
f_3	mean	2.470e+05 +	1.102e+05 +	9.569e+04 +	8.408e+04 +	9.200e+04 +	5.927e+04 −	**2.175e+04** −	2.655e+04 −	2.642e+04 −	1.193e+05 +	6.926e+04 +	6.730e+04
	std	3.864e+04	2.516e+04	1.468e+04	1.644e+04	1.334e+04	7.617e+03	3.030e+03	8.795e+03	6.266e+03	1.705e+04	1.179e+04	1.113e+04
f_4	mean	3.562e+03 +	2.313e+03 +	7.315e+02 +	**4.940e+02** −	5.099e+02 ≈	5.191e+02 +	6.448e+02 +	5.722e+02 +	5.666e+02 +	8.868e+02 +	5.159e+02 +	5.153e+02
	std	6.544e+02	1.346e+03	5.634e+01	1.249e+01	1.367e+01	2.621e+00	7.635e+01	3.779e+01	3.238e+01	1.051e+02	3.825e+00	2.593e+00
f_5	mean	8.913e+02 +	8.408e+02 +	7.709e+02 +	**6.209e+02** −	6.493e+02 −	6.703e+02 ≈	8.029e+02 +	6.714e+02 ≈	6.741e+02 ≈	7.641e+02 +	6.707e+02 ≈	6.681e+02
	std	2.249e+01	3.818e+01	1.169e+01	1.631e+01	1.171e+01	6.441e+00	5.603e+00	3.247e+01	2.237e+01	2.444e+01	7.149e+00	9.385e+00
f_6	mean	6.752e+02 +	6.612e+02 +	6.397e+02 +	6.003e+02 +	6.030e+02 +	6.008e+02 +	6.650e+02 +	6.102e+02 +	6.091e+02 +	6.379e+02 +	6.000e+02 ≈	**6.000e+02**
	std	2.929e+00	1.112e+01	5.025e+00	3.884e-01	5.588e-01	9.026e-02	1.112e+00	2.900e+00	2.090e+00	8.409e+00	8.144e-04	7.524e-04
f_7	mean	2.491e+03 +	1.127e+03 +	1.171e+03 +	**8.787e+02** −	8.831e+02 −	9.057e+02 ≈	1.312e+03 +	9.724e+02 +	9.750e+02 +	1.249e+03 +	8.974e+02 ≈	9.027e+02
	std	1.734e+02	5.996e+01	3.224e+01	2.049e+01	1.176e+01	8.960e+00	1.274e+01	2.081e+01	3.268e+01	8.887e+01	9.571e+00	1.002e+01
f_8	mean	1.189e+03 +	1.138e+03 +	1.076e+03 +	**9.201e+02** −	9.491e+02 −	9.659e+02 ≈	9.965e+02 +	9.805e+02 +	9.819e+02 ≈	1.062e+03 +	9.654e+02 −	9.728e+02
	std	1.371e+01	3.069e+01	1.286e+01	1.013e+01	8.715e+00	1.031e+01	5.381e+00	2.351e+01	1.611e+01	2.308e+01	1.179e+01	9.299e+00
f_9	mean	1.711e+04 +	8.930e+03 +	5.133e+03 +	9.001e+02 +	2.360e+03 +	9.014e+02 +	6.436e+03 +	1.403e+03 +	1.287e+03 +	6.420e+03 +	**9.000e+02** −	9.000e+02
	std	1.973e+03	2.484e+03	1.085e+03	1.974e-01	3.802e+02	1.828e-01	7.223e+02	1.851e+02	1.223e+02	1.127e+03	1.407e-05	1.456e-05
f_{10}	mean	8.016e+03 ≈	8.783e+03 +	8.618e+03 +	8.063e+03 ≈	6.170e+03 −	7.671e+03 −	**5.227e+03** −	8.468e+03 +	8.430e+03 +	7.167e+03 −	8.241e+03 +	8.127e+03
	std	3.922e+02	2.907e+02	3.571e+02	5.985e+02	2.960e+02	2.527e+02	2.032e+02	2.848e+02	2.909e+02	5.887e+02	3.624e+02	4.236e+02
f_{11}	mean	3.139e+03 +	4.083e+03 +	1.354e+03 +	1.240e+03 ≈	1.318e+03 +	1.302e+03 +	**1.204e+03** ≈	1.308e+03 +	1.283e+03 +	4.551e+03 +	1.206e+03 ≈	1.209e+03
	std	3.719e+02	2.127e+03	2.502e+01	7.731e+01	1.776e+01	2.836e+01	2.446e+01	4.649e+01	3.995e+01	1.445e+03	3.486e+01	3.203e+01
f_{12}	mean	1.607e+09 +	5.926e+08 +	2.176e+07 +	**7.420e+04** −	1.517e+06 +	1.120e+06 +	8.632e+05 ≈	2.948e+06 +	2.474e+06 +	1.469e+08 +	7.643e+05 ≈	6.332e+05
	std	3.076e+08	7.102e+08	3.724e+06	7.377e+04	3.619e+05	4.482e+05	6.164e+05	2.354e+06	2.837e+06	6.443e+07	4.226e+05	3.376e+05
f_{13}	mean	1.126e+08 +	8.453e+07 +	**7.858e+03** −	4.456e+04 +	1.499e+05 +	1.538e+04 +	1.833e+04 +	2.395e+04 +	2.313e+04 +	4.819e+06 +	1.156e+04 +	1.119e+04
	std	4.749e+07	2.695e+08	1.223e+03	1.740e+05	6.297e+04	4.312e+03	8.058e+03	1.922e+04	1.777e+04	4.974e+06	4.148e+03	4.301e+03
f_{14}	mean	1.039e+04 −	7.256e+05 +	**1.506e+03** −	4.692e+04 −	1.456e+04 −	2.482e+05 +	1.618e+03 −	1.564e+03 −	1.552e+03 −	1.314e+05 ≈	1.419e+05 ≈	1.369e+05
	std	3.458e+03	5.866e+05	9.434e+00	7.349e+04	1.271e+04	8.973e+04	3.068e+02	4.451e+01	3.091e+01	8.450e+04	9.784e+04	1.100e+05
f_{15}	mean	4.774e+05 +	1.422e+06 +	**1.820e+03** ≈	4.260e+04 ≈	3.185e+04 +	2.658e+03 ≈	1.905e+03 ≈	2.508e+03 ≈	2.487e+03 ≈	3.754e+05 +	2.337e+03 +	2.296e+03
	std	1.782e+05	4.954e+06	4.750e+01	8.582e+04	1.434e+04	6.952e+02	2.179e+02	5.402e+02	4.660e+02	6.154e+05	9.627e+02	7.897e+02
f_{16}	mean	3.809e+03 +	4.262e+03 +	3.573e+03 +	2.806e+03 ≈	2.741e+03 ≈	**2.608e+03** −	3.060e+03 +	2.937e+03 ≈	2.873e+03 ≈	3.182e+03 +	2.828e+03 ≈	2.799e+03
	std	2.242e+02	2.997e+02	1.416e+02	2.338e+02	1.023e+02	3.273e+02	2.485e+02	1.889e+02	2.610e+02	2.724e+02	2.559e+02	3.065e+02
f_{17}	mean	2.733e+03 +	2.909e+03 +	2.620e+03 +	2.097e+03 +	2.027e+03 +	1.924e+03 +	1.992e+03 +	2.084e+03 +	2.093e+03 +	2.347e+03 +	1.859e+03 +	**1.851e+03**
	std	1.897e+02	2.367e+02	1.018e+02	1.353e+02	7.430e+01	6.196e+01	1.844e+02	1.100e+02	1.029e+02	1.373e+02	8.266e+01	8.824e+01
f_{18}	mean	5.041e+06 +	1.142e+07 +	**4.276e+03** −	8.762e+04 −	3.878e+05 ≈	4.268e+05 ≈	2.841e+04 −	2.082e+04 −	1.677e+04 −	3.089e+06 +	5.208e+05 ≈	6.238e+05
	std	2.491e+06	9.287e+06	1.420e+03	2.609e+05	1.939e+05	2.183e+05	1.436e+04	2.064e+04	1.046e+04	1.746e+06	4.701e+05	5.385e+05
f_{19}	mean	9.226e+06 +	1.971e+07 +	**2.138e+03** −	9.532e+03 +	2.673e+04 +	5.575e+03 ≈	3.435e+03 −	2.727e+03 −	2.236e+03 −	6.244e+05 +	5.096e+03 ≈	4.885e+03
	std	4.048e+06	6.543e+07	7.751e+01	3.258e+04	1.407e+04	1.972e+03	2.054e+03	2.450e+03	3.370e+02	7.343e+05	1.690e+03	1.431e+03
f_{20}	mean	2.600e+03 +	2.935e+03 +	2.930e+03 +	2.496e+03 +	2.412e+03 +	2.323e+03 +	3.107e+03 +	2.559e+03 +	2.542e+03 +	2.556e+03 +	2.336e+03 +	**2.255e+03**
	std	1.902e+02	1.149e+02	9.758e+01	1.233e+02	9.401e+01	6.259e+01	2.650e+01	1.361e+02	1.590e+02	1.131e+02	1.289e+02	9.936e+01
f_{21}	mean	2.666e+03 +	2.623e+03 +	2.561e+03 +	**2.416e+03** −	2.441e+03 −	2.458e+03 ≈	2.662e+03 +	2.459e+03 ≈	2.466e+03 ≈	2.545e+03 +	2.457e+03 ≈	2.460e+03
	std	1.769e+01	4.452e+01	1.423e+01	1.004e+01	1.060e+01	1.056e+01	5.660e+01	3.087e+01	1.901e+01	2.344e+01	7.265e+00	9.664e+00
f_{22}	mean	9.845e+03 +	3.862e+03 +	9.808e+03 +	3.179e+03 +	2.619e+03 +	2.313e+03 +	7.707e+03 +	2.535e+03 +	2.553e+03 +	3.422e+03 +	2.300e+03 +	**2.300e+03**
	std	2.418e+02	2.089e+03	1.083e+03	2.173e+03	1.447e+02	5.433e-01	2.075e+02	8.276e+01	8.110e+01	8.176e+02	1.334e-03	1.427e-03
f_{23}	mean	2.997e+03 +	3.059e+03 +	2.915e+03 +	2.770e+03 +	2.793e+03 +	2.794e+03 +	4.520e+03 +	2.795e+03 +	2.788e+03 +	2.908e+03 +	2.731e+03 +	**2.710e+03**
	std	1.760e+01	7.446e+01	1.356e+01	1.602e+01	1.114e+01	1.730e+01	1.656e+02	2.758e+01	3.480e+01	2.451e+01	5.889e+01	5.597e+01
f_{24}	mean	3.128e+03 +	3.265e+03 +	3.080e+03 +	2.944e+03 ≈	2.967e+03 ≈	2.957e+03 ≈	3.035e+03 ≈	2.979e+03 +	2.968e+03 ≈	3.061e+03 +	2.924e+03 +	**2.933e+03**
	std	1.504e+01	6.563e+01	1.278e+01	1.609e+01	8.009e+00	2.442e+01	2.085e+02	2.654e+01	3.010e+01	1.932e+01	6.001e+01	5.846e+01
f_{25}	mean	6.435e+03 +	3.233e+03 +	3.151e+03 +	**2.887e+03** −	2.891e+03 +	2.891e+03 +	2.982e+03 +	3.006e+03 +	2.999e+03 +	3.284e+03 +	2.888e+03 ≈	2.889e+03
	std	5.555e+02	2.169e+02	4.868e+01	3.769e-01	4.364e+00	1.726e+00	1.918e+01	3.813e+01	3.103e+01	9.551e+01	1.672e+00	2.083e+00
f_{26}	mean	7.518e+03 +	7.806e+03 +	6.346e+03 +	4.738e+03 +	5.093e+03 +	4.691e+03 +	6.867e+03 +	5.127e+03 +	5.192e+03 +	6.511e+03 +	3.767e+03 ≈	**3.751e+03**
	std	2.751e+02	6.445e+02	1.820e+02	1.190e+02	1.132e+02	1.803e+02	1.299e+03	4.150e+02	2.712e+02	2.340e+02	1.108e+02	2.828e+02
f_{27}	mean	3.284e+03 +	3.691e+03 +	3.242e+03 +	**3.212e+03** −	3.231e+03 +	3.226e+03 +	6.542e+03 +	3.224e+03 ≈	3.227e+03 +	3.317e+03 +	3.220e+03 ≈	3.219e+03
	std	1.156e+01	1.062e+02	1.116e+01	5.608e+00	4.003e+00	5.023e+00	2.294e+02	1.123e+01	1.121e+01	2.628e+01	9.583e+00	6.371e+00
f_{28}	mean	5.020e+03 +	4.136e+03 +	3.470e+03 +	**3.216e+03** −	3.246e+03 ≈	3.268e+03 +	3.355e+03 +	3.334e+03 +	3.344e+03 +	3.602e+03 +	3.239e+03 +	3.236e+03
	std	9.307e+02	4.846e+02	5.533e+01	1.491e+01	1.861e+01	1.298e+01	4.013e+01	5.577e+01	4.377e+01	1.144e+02	1.493e+01	1.433e+01
f_{29}	mean	4.276e+03 +	5.487e+03 +	4.448e+03 +	3.732e+03 +	3.748e+03 +	3.645e+03 +	3.727e+03 +	3.895e+03 +	3.893e+03 +	4.474e+03 +	3.500e+03 +	**3.490e+03**
	std	3.301e+02	3.644e+02	1.464e+02	8.683e+01	8.185e+01	7.962e+01	1.581e+02	1.421e+02	1.579e+02	1.778e+02	8.280e+01	8.217e+01
f_{30}	mean	1.045e+07 +	2.640e+07 +	8.553e+04 +	1.609e+04 ≈	1.983e+05 +	3.585e+04 +	3.811e+04 +	5.946e+04 +	6.096e+04 +	2.892e+06 +	8.428e+03 +	**8.474e+03**
	std	2.938e+06	3.504e+07	2.641e+04	2.644e+04	8.131e+04	1.030e+04	4.053e+04	4.763e+04	5.098e+04	2.061e+06	1.544e+03	1.590e+03
+/≈/−:		27/1/1	29/0/0	24/1/4	11/7/11	18/5/6	18/8/3	20/4/5	20/5/4	19/6/4	27/1/1	14/13/2	-
Avg. ranks:		10.79	11.03	7.83	3.93	5.21	4.59	6.90	6.10	5.79	9.24	3.41	**3.17**

Table 12
Detailed experimental results on 50-D CEC2017.

Func.		DE	PSO	CMA-ES	JADE	L-SHADE	CSO	jSO	MCSO	SCSO	WLCSO	FDBCSO	DFDBCSO
f_1	mean	1.049e+11 +	1.868e+10 +	1.021e+10 +	3.237e+03 +	**7.926e+02** ≈	1.530e+05 +	3.082e+08 +	1.525e+10 +	1.570e+10 +	1.390e+10 +	8.549e+02 ≈	1.328e+03
	std	1.061e+10	3.852e+09	2.700e+09	3.516e+03	6.155e+02	2.172e+04	1.685e+08	3.347e+09	3.960e+09	3.261e+09	9.612e+02	1.747e+03
f_3	mean	4.672e+05 +	2.307e+05 +	2.506e+05 +	1.873e+05 +	1.658e+05 +	1.242e+05 −	**6.966e+04** −	8.013e+04 −	7.975e+04 −	2.478e+05 +	1.509e+05 ≈	1.500e+05
	std	5.827e+04	3.702e+04	2.225e+04	4.606e+04	6.381e+04	1.623e+04	1.185e+04	1.284e+04	1.547e+04	2.837e+04	1.714e+04	1.609e+04
f_4	mean	1.504e+04 +	5.264e+03 +	1.638e+03 +	5.398e+02 −	**5.275e+02** −	5.896e+02 ≈	7.116e+02 +	1.533e+03 +	1.279e+03 +	1.979e+03 +	5.700e+02 ≈	5.790e+02
	std	2.802e+03	1.564e+03	2.660e+02	4.296e+01	3.864e+01	3.292e+01	6.978e+01	3.418e+02	2.470e+02	5.345e+02	3.811e+01	3.115e+01
f_5	mean	1.283e+03 +	1.025e+03 +	1.016e+03 +	**7.151e+02** −	7.906e+02 −	8.069e+02 −	8.753e+02 +	7.790e+02 −	7.813e+02 −	1.020e+03 +	8.232e+02 ≈	8.323e+02
	std	3.013e+01	7.610e+01	1.727e+01	2.177e+01	1.817e+01	1.710e+01	9.125e+00	5.718e+01	6.215e+01	4.164e+01	1.668e+01	1.607e+01
f_6	mean	6.970e+02 +	6.770e+02 +	6.535e+02 +	6.001e+02 +	6.015e+02 +	6.003e+02 +	6.696e+02 +	6.198e+02 +	6.189e+02 +	6.494e+02 +	6.000e+02 ≈	**6.000e+02**
	std	3.711e+00	1.497e+01	3.503e+00	1.157e-01	4.060e-01	2.102e-02	7.623e-01	3.278e+00	3.001e+00	8.077e+00	9.400e-05	6.772e-05
f_7	mean	4.851e+03 +	1.622e+03 +	1.510e+03 +	**1.002e+03** −	1.037e+03 −	1.074e+03 ≈	1.766e+03 +	1.303e+03 +	1.344e+03 +	1.785e+03 +	1.081e+03 ≈	1.081e+03
	std	3.777e+02	1.234e+02	6.948e+01	3.039e+01	1.520e+01	1.428e+01	1.458e+01	6.849e+01	8.146e+01	1.378e+02	1.442e+01	1.165e+01
f_8	mean	1.576e+03 +	1.365e+03 +	1.313e+03 +	**1.022e+03** −	1.090e+03 −	1.104e+03 −	1.217e+03 +	1.063e+03 −	1.036e+03 −	1.312e+03 +	1.126e+03 ≈	1.128e+03
	std	4.417e+01	1.151e+02	2.027e+01	1.658e+01	1.412e+01	1.565e+01	6.975e+00	5.071e+01	2.525e+01	2.843e+01	1.665e+01	1.726e+01
f_9	mean	4.855e+04 +	3.165e+04 +	1.338e+04 +	9.045e+02 +	7.625e+03 +	9.007e+02 +	1.544e+04 +	3.876e+03 +	4.437e+03 +	1.619e+04 +	9.004e+02 ≈	**9.006e+02**
	std	4.034e+03	9.697e+03	3.951e+03	2.711e+00	1.163e+03	1.103e-01	1.135e+03	8.501e+02	1.101e+03	3.877e+03	4.615e-01	8.970e-01
f_{10}	mean	1.485e+04 +	1.508e+04 +	1.522e+04 +	1.423e+04 ≈	1.011e+04 −	1.204e+04 −	**8.082e+03** −	1.470e+04 +	1.457e+04 +	1.282e+04 −	1.422e+04 ≈	1.413e+04
	std	5.364e+02	3.328e+02	2.796e+02	4.521e+02	4.111e+02	7.562e+02	3.747e+02	3.746e+02	5.331e+02	7.610e+02	4.774e+02	3.922e+02
f_{11}	mean	2.231e+04 +	1.285e+04 +	1.629e+03 −	1.589e+03 −	1.769e+03 ≈	2.197e+03 +	**1.392e+03** −	1.908e+03 ≈	1.860e+03 ≈	1.266e+04 +	1.841e+03 ≈	1.889e+03
	std	3.984e+03	5.506e+03	6.374e+01	8.914e+02	1.820e+02	3.181e+02	5.813e+01	1.953e+02	2.112e+02	2.095e+03	3.841e+02	3.674e+02
f_{12}	mean	1.340e+10 +	3.399e+09 +	2.686e+08 +	**3.199e+05** −	3.434e+06 +	2.924e+06 ≈	5.830e+06 +	2.080e+08 +	2.610e+08 +	8.024e+08 +	2.554e+06 ≈	2.601e+06
	std	2.160e+09	2.315e+09	5.988e+07	2.169e+05	1.217e+06	1.204e+06	2.671e+06	1.404e+08	2.529e+08	3.511e+08	7.096e+05	1.328e+06
f_{13}	mean	2.260e+09 +	1.154e+08 +	2.122e+05 +	7.653e+03 +	5.350e+04 +	2.741e+03 +	1.551e+04 +	6.391e+04 +	6.490e+04 +	1.468e+07 +	2.448e+03 +	**2.314e+03**
	std	5.706e+08	2.384e+08	5.889e+04	5.295e+03	2.406e+04	5.923e+02	6.259e+03	2.738e+04	3.248e+04	1.397e+07	1.029e+03	9.629e+02
f_{14}	mean	1.657e+06 +	7.745e+06 +	**1.621e+03** −	4.532e+04 −	5.629e+04 −	4.312e+05 +	5.156e+03 −	1.848e+03 −	1.738e+03 −	1.049e+06 +	3.807e+05 ≈	3.528e+05
	std	7.885e+05	5.761e+06	1.599e+01	1.341e+05	1.053e+05	1.559e+05	5.106e+03	5.844e+02	7.660e+01	4.962e+05	2.375e+05	2.997e+05
f_{15}	mean	4.005e+07 +	1.336e+06 +	**3.619e+03** −	1.512e+04 +	2.926e+04 +	5.058e+03 ≈	1.133e+04 +	6.012e+03 ≈	6.356e+03 ≈	1.126e+05 +	5.392e+03 ≈	5.427e+03
	std	2.267e+07	3.431e+06	4.039e+02	5.661e+04	1.467e+04	1.755e+03	3.997e+03	2.545e+03	2.277e+03	5.287e+04	2.290e+03	2.336e+03
f_{16}	mean	6.154e+03 +	6.258e+03 +	5.435e+03 +	3.916e+03 +	3.547e+03 +	2.479e+03 ≈	3.052e+03 +	3.013e+03 +	2.919e+03 +	4.882e+03 +	2.760e+03 +	**2.315e+03**
	std	3.342e+02	6.864e+02	2.643e+02	3.371e+02	2.415e+02	4.485e+02	1.919e+02	4.500e+02	4.875e+02	3.661e+02	7.545e+02	4.465e+02
f_{17}	mean	5.079e+03 +	4.825e+03 +	4.250e+03 +	3.191e+03 −	3.001e+03 −	**2.851e+03** −	2.879e+03 −	3.108e+03 ≈	3.387e+03 ≈	4.027e+03 +	3.313e+03 ≈	3.273e+03
	std	2.143e+02	3.651e+02	1.673e+02	1.767e+02	1.184e+02	3.220e+02	2.436e+02	4.973e+02	4.690e+02	2.777e+02	3.272e+02	4.358e+02
f_{18}	mean	1.981e+07 +	5.962e+07 +	5.707e+04 −	**1.766e+04** −	2.177e+06 ≈	2.211e+06 ≈	1.220e+05 −	7.321e+04 −	5.605e+04 −	1.040e+07 +	2.051e+06 ≈	2.265e+06
	std	6.568e+06	3.701e+07	1.256e+04	6.832e+03	6.910e+05	5.787e+05	8.631e+04	6.647e+04	2.609e+04	5.487e+06	6.232e+05	6.681e+05
f_{19}	mean	4.577e+07 +	1.111e+07 +	5.182e+03 −	**2.094e+03** −	2.195e+04 +	1.599e+04 ≈	1.678e+04 ≈	1.295e+04 −	1.167e+04 ≈	2.079e+05 +	1.569e+04 ≈	1.548e+04
	std	1.477e+07	3.858e+07	7.688e+02	4.106e+01	4.970e+03	2.741e+03	3.175e+03	1.403e+04	8.410e+03	4.335e+05	2.672e+03	3.493e+03
f_{20}	mean	4.383e+03 +	4.235e+03 +	4.247e+03 +	3.417e+03 +	3.225e+03 ≈	**2.458e+03** −	3.671e+03 +	3.711e+03 +	3.688e+03 +	3.529e+03 +	3.232e+03 ≈	3.162e+03
	std	1.962e+02	1.396e+02	1.267e+02	2.192e+02	1.257e+02	2.328e+02	7.749e+01	1.321e+02	2.656e+02	2.186e+02	3.718e+02	3.656e+02
f_{21}	mean	3.074e+03 +	2.942e+03 +	2.821e+03 +	**2.515e+03** −	2.576e+03 −	2.588e+03 −	2.800e+03 +	2.533e+03 −	2.540e+03 −	2.796e+03 +	2.625e+03 ≈	2.624e+03
	std	3.382e+01	1.176e+02	2.771e+01	1.513e+01	1.789e+01	1.909e+01	5.513e+01	2.911e+01	4.616e+01	4.740e+01	3.819e+01	2.004e+01
f_{22}	mean	1.670e+04 +	1.659e+04 +	1.660e+04 +	1.532e+04 +	1.019e+04 +	**2.311e+03** −	1.290e+04 +	1.478e+04 +	1.342e+04 +	1.404e+04 +	4.849e+03 ≈	4.847e+03
	std	2.790e+02	3.406e+02	4.041e+02	2.233e+03	2.688e+03	5.998e+00	3.064e+02	3.720e+03	4.901e+03	1.795e+03	5.090e+03	5.096e+03
f_{23}	mean	3.496e+03 +	3.654e+03 +	3.315e+03 +	2.945e+03 +	3.019e+03 +	2.893e+03 +	5.125e+03 +	2.958e+03 +	2.986e+03 +	3.283e+03 +	**2.775e+03** −	2.821e+03
	std	3.402e+01	8.767e+01	4.790e+01	1.851e+01	1.680e+01	4.285e+01	1.704e+02	3.007e+01	5.134e+01	5.319e+01	5.020e+01	9.077e+01
f_{24}	mean	3.560e+03 +	3.866e+03 +	3.480e+03 +	3.118e+03 +	3.181e+03 +	**2.956e+03** −	3.107e+03 +	3.096e+03 +	3.128e+03 +	3.397e+03 +	2.960e+03 ≈	2.982e+03
	std	2.658e+01	1.405e+02	4.342e+01	2.170e+01	1.710e+01	2.811e+01	7.945e+01	3.013e+01	6.069e+01	4.180e+01	8.283e+01	1.124e+02
f_{25}	mean	1.758e+04 +	5.128e+03 +	4.070e+03 +	**3.032e+03** −	3.095e+03 ≈	3.091e+03 ≈	3.265e+03 +	4.078e+03 +	4.272e+03 +	4.274e+03 +	3.090e+03 +	3.087e+03
	std	1.945e+03	5.006e+02	3.009e+02	2.331e+01	1.958e+01	1.033e+01	5.535e+01	2.371e+02	6.684e+02	3.805e+02	1.123e+01	1.478e+01
f_{26}	mean	1.170e+04 +	1.283e+04 +	9.213e+03 +	5.768e+03 +	6.532e+03 +	4.365e+03 +	1.247e+04 +	6.189e+03 +	6.303e+03 +	9.224e+03 +	4.131e+03 +	**4.041e+03**
	std	3.476e+02	1.303e+03	3.675e+02	2.488e+02	1.969e+02	1.965e+02	8.692e+02	3.080e+02	4.028e+02	3.314e+02	2.163e+02	1.280e+02
f_{27}	mean	3.662e+03 +	5.173e+03 +	3.493e+03 +	**3.304e+03** −	3.419e+03 +	3.331e+03 ≈	1.037e+04 +	3.399e+03 +	3.406e+03 +	3.797e+03 +	3.347e+03 ≈	3.342e+03
	std	8.222e+01	3.421e+02	9.810e+01	4.195e+01	2.298e+01	1.761e+01	1.473e+03	5.924e+01	5.240e+01	1.233e+02	3.116e+01	3.163e+01
f_{28}	mean	8.864e+03 +	6.638e+03 +	3.917e+03 +	**3.311e+03** −	3.366e+03 ≈	3.377e+03 ≈	3.693e+03 +	4.511e+03 +	4.491e+03 +	5.347e+03 +	3.368e+03 ≈	3.370e+03
	std	2.295e+02	1.601e+03	2.030e+02	6.730e+00	1.970e+01	1.025e+01	7.163e+01	5.559e+02	3.720e+02	7.908e+02	2.101e+01	1.685e+01
f_{29}	mean	6.850e+03 +	8.252e+03 +	5.939e+03 +	4.044e+03 +	4.100e+03 +	3.778e+03 +	4.190e+03 +	4.139e+03 +	4.091e+03 +	6.526e+03 +	3.521e+03 +	**3.443e+03**
	std	3.133e+02	1.438e+03	1.381e+02	1.861e+02	1.279e+02	1.970e+02	1.868e+02	2.778e+02	2.083e+02	4.724e+02	1.664e+02	1.617e+02
f_{30}	mean	3.349e+08 +	3.609e+08 +	9.572e+06 +	**8.198e+05** −	4.525e+06 +	1.093e+06 +	2.161e+06 +	7.128e+06 +	7.173e+06 +	5.324e+07 +	9.068e+05 ≈	8.929e+05
	std	8.547e+07	4.556e+08	2.134e+06	1.256e+05	7.610e+05	9.097e+04	4.876e+05	3.605e+06	3.490e+06	1.736e+07	5.135e+04	5.667e+04
+/≈/−:		29/0/0	29/0/0	24/0/5	13/1/15	15/6/8	10/10/9	22/1/6	19/3/7	19/4/6	28/0/1	5/23/1	-
Avg. ranks:		11.48	11.03	7.93	**3.55**	5.10	3.93	6.52	5.79	5.86	9.14	3.83	3.83

Table 13
Detailed experimental results on 100-D CEC2017.

Func.		DE	PSO	CMA-ES	JADE	L-SHADE	CSO	jSO	MCSO	SCSO	WLCSO	FDBCSO	DFDBCSO
f_1	mean	3.306e+11 +	9.590e+10 +	8.272e+10 +	**3.647e+03** ≈	4.860e+03 ≈	5.640e+03 +	1.301e+09 +	1.134e+11 +	1.173e+11 +	4.010e+10 +	3.848e+03 ≈	4.627e+03
	std	1.994e+10	9.214e+09	1.300e+10	4.023e+03	2.727e+03	2.475e+03	2.840e+08	1.668e+10	1.479e+10	6.254e+09	4.097e+03	2.858e+03
f_3	mean	1.119e+06 +	5.653e+05 +	6.732e+05 +	4.800e+05 +	4.875e+05 +	2.903e+05 −	**2.080e+05** −	2.604e+05 −	2.643e+05 −	6.204e+05 +	3.763e+05 ≈	3.816e+05
	std	9.720e+04	5.919e+04	7.860e+04	5.579e+04	4.247e+04	2.448e+04	1.479e+04	2.876e+04	3.069e+04	5.696e+04	2.824e+04	3.016e+04
f_4	mean	7.529e+04 +	2.023e+04 +	9.585e+03 +	**6.436e+02** ≈	6.994e+02 ≈	7.476e+02 +	1.186e+03 +	1.239e+04 +	1.166e+04 +	5.366e+03 +	7.250e+02 ≈	7.399e+02
	std	9.909e+03	3.898e+03	1.593e+03	4.170e+01	3.816e+01	2.557e+01	9.749e+01	3.108e+03	2.991e+03	1.201e+03	2.835e+01	3.960e+01
f_5	mean	2.303e+03 +	1.674e+03 +	1.685e+03 +	1.032e+03 −	1.209e+03 −	**6.497e+02** −	1.345e+03 +	1.360e+03 +	1.357e+03 +	1.662e+03 +	1.270e+03 ≈	1.248e+03
	std	6.890e+01	1.295e+02	4.651e+01	5.710e+01	3.992e+01	4.916e+01	1.334e+01	5.548e+01	5.988e+01	8.237e+01	1.538e+02	1.031e+02
f_6	mean	7.240e+02 +	6.859e+02 +	6.681e+02 +	6.011e+02 +	6.002e+02 +	6.000e+02 +	6.661e+02 +	6.371e+02 +	6.374e+02 +	6.597e+02 +	6.000e+02 ≈	**6.000e+02**
	std	5.374e+00	1.493e+01	5.320e+00	5.157e-01	7.458e-02	3.514e-03	1.096e+00	3.643e+00	5.175e+00	5.398e+00	1.133e-03	4.342e-06
f_7	mean	1.159e+04 +	3.203e+03 +	2.763e+03 +	**1.466e+03** −	1.538e+03 −	1.507e+03 −	3.216e+03 +	2.832e+03 +	2.816e+03 +	3.253e+03 +	1.566e+03 ≈	1.571e+03
	std	6.179e+02	2.672e+02	1.375e+02	6.772e+01	3.369e+01	3.094e+01	2.017e+01	1.648e+02	1.465e+02	1.846e+02	1.513e+01	1.242e+01
f_8	mean	2.674e+03 +	2.070e+03 +	1.994e+03 +	1.344e+03 −	1.499e+03 −	**9.326e+02** −	1.896e+03 +	1.637e+03 +	1.630e+03 +	1.983e+03 +	1.547e+03 +	1.585e+03
	std	6.786e+01	1.648e+02	4.940e+01	5.523e+01	2.539e+01	7.606e+01	9.110e+00	5.441e+01	5.559e+01	8.566e+01	1.419e+02	3.322e+01
f_9	mean	1.355e+05 +	8.296e+04 +	4.854e+04 +	2.686e+03 +	3.207e+04 +	**9.014e+02** −	2.609e+04 +	2.075e+04 +	2.083e+04 +	4.869e+04 +	9.350e+02 +	9.179e+02
	std	1.128e+04	1.914e+04	7.634e+03	1.286e+03	4.702e+03	1.080e+00	5.466e+02	2.987e+03	3.214e+03	6.318e+03	3.021e+01	8.418e+00
f_{10}	mean	3.262e+04 +	3.220e+04 +	3.207e+04 +	3.110e+04 +	2.165e+04 −	1.927e+04 −	**1.623e+04** −	3.128e+04 +	3.108e+04 +	2.857e+04 −	3.017e+04 ≈	3.024e+04
	std	5.340e+02	5.911e+02	4.488e+02	5.678e+02	6.921e+02	2.441e+03	3.247e+02	9.972e+02	7.017e+02	1.178e+03	6.974e+02	6.848e+02
f_{11}	mean	4.133e+05 +	1.923e+05 +	1.160e+04 ≈	1.105e+04 −	4.956e+04 +	1.918e+04 +	**6.971e+03** −	4.143e+04 +	3.987e+04 +	2.039e+05 +	1.287e+04 +	1.133e+04
	std	5.344e+04	2.770e+04	2.554e+03	2.059e+04	9.489e+03	3.248e+03	1.377e+03	1.176e+04	1.319e+04	3.099e+04	2.985e+03	2.824e+03
f_{12}	mean	8.924e+10 +	3.622e+10 +	3.882e+09 +	**1.129e+06** −	8.615e+06 −	1.633e+07 +	6.926e+07 +	1.030e+10 +	1.068e+10 +	3.989e+09 +	1.284e+07 ≈	1.385e+07
	std	1.136e+10	9.239e+09	1.223e+09	5.120e+05	3.560e+06	4.514e+06	1.900e+07	3.746e+09	2.432e+09	1.057e+09	4.438e+06	3.893e+06
f_{13}	mean	9.028e+09 +	1.437e+09 +	1.051e+07 +	5.435e+03 ≈	4.404e+03 ≈	4.488e+03 ≈	1.197e+04 +	4.400e+08 +	4.545e+08 +	4.200e+07 +	4.239e+03 ≈	**3.692e+03**
	std	1.040e+09	9.442e+08	2.386e+06	3.649e+03	1.513e+03	1.098e+03	1.967e+03	2.756e+08	2.188e+08	2.738e+07	2.433e+03	1.503e+03
f_{14}	mean	2.783e+07 +	5.353e+07 +	**2.716e+03** −	4.347e+04 −	8.213e+05 −	1.899e+06 ≈	1.992e+05 −	1.033e+05 −	7.966e+04 −	9.120e+06 +	1.773e+06 +	1.613e+06
	std	1.477e+07	4.527e+07	1.166e+02	3.816e+04	1.147e+06	3.870e+05	7.254e+04	1.107e+05	6.331e+04	5.809e+06	2.965e+05	3.744e+05
f_{15}	mean	1.255e+09 +	9.059e+07 +	3.991e+04 +	4.627e+03 +	5.360e+03 +	**2.106e+03** ≈	4.394e+03 +	1.173e+06 +	4.293e+06 +	1.142e+06 +	2.121e+03 +	2.079e+03
	std	4.688e+08	1.810e+08	7.929e+03	3.446e+03	2.236e+03	3.307e+02	1.149e+03	1.830e+06	7.318e+06	2.636e+06	4.570e+02	3.381e+02
f_{16}	mean	1.346e+04 +	1.264e+04 +	1.119e+04 +	7.069e+03 +	6.804e+03 +	**2.984e+03** ≈	6.228e+03 +	5.985e+03 +	5.975e+03 +	1.011e+04 +	3.370e+03 +	3.290e+03
	std	4.297e+02	1.671e+03	2.523e+02	8.867e+02	4.285e+02	3.173e+02	3.154e+02	7.785e+02	6.087e+02	8.220e+02	1.173e+03	1.293e+03
f_{17}	mean	1.300e+04 +	9.932e+03 +	7.779e+03 +	5.821e+03 ≈	5.125e+03 ≈	**2.782e+03** −	6.145e+03 +	4.334e+03 −	4.410e+03 −	7.345e+03 +	5.275e+03 +	4.915e+03
	std	1.192e+03	4.535e+03	3.292e+02	5.671e+02	2.233e+02	2.864e+02	3.522e+02	4.643e+02	5.641e+02	6.398e+02	1.131e+03	1.390e+03
f_{18}	mean	8.548e+07 +	8.689e+07 +	5.132e+05 −	**1.276e+05** −	2.407e+06 ≈	2.214e+06 ≈	4.628e+05 −	2.562e+05 −	3.014e+05 −	2.714e+07 +	2.363e+06 +	2.171e+06
	std	3.582e+07	5.880e+07	1.421e+05	5.272e+04	1.222e+06	5.723e+05	1.782e+05	1.054e+05	2.309e+05	1.248e+07	5.082e+05	4.887e+05
f_{19}	mean	1.899e+09 +	1.412e+08 +	3.674e+05 +	4.849e+03 ≈	4.136e+03 +	**2.788e+03** ≈	3.533e+03 +	8.416e+06 +	9.975e+06 +	2.213e+06 +	3.077e+03 +	2.756e+03
	std	4.689e+08	2.090e+08	1.211e+05	4.228e+03	1.562e+03	8.288e+02	6.599e+02	1.011e+07	1.065e+07	2.078e+06	1.294e+03	4.443e+02
f_{20}	mean	8.040e+03 +	7.807e+03 +	7.624e+03 +	6.855e+03 +	5.712e+03 −	**2.692e+03** −	6.250e+03 ≈	6.660e+03 +	6.745e+03 +	6.265e+03 ≈	6.062e+03 ≈	6.155e+03
	std	3.648e+02	2.231e+02	3.129e+02	3.292e+02	2.194e+02	2.202e+02	4.692e+01	3.740e+02	3.633e+02	3.159e+02	8.042e+02	5.324e+02
f_{21}	mean	4.248e+03 +	3.908e+03 +	3.641e+03 +	2.837e+03 −	2.984e+03 +	2.397e+03 −	3.180e+03 +	3.138e+03 +	**3.102e+03** ≈	3.534e+03 +	2.966e+03 +	2.908e+03
	std	8.025e+01	1.830e+02	6.147e+01	4.147e+01	3.401e+01	3.430e+01	4.627e+01	6.208e+01	4.840e+01	5.654e+01	2.373e+02	2.774e+02
f_{22}	mean	3.455e+04 +	3.448e+04 +	3.426e+04 +	3.305e+04 +	2.384e+04 −	**1.295e+04** −	2.180e+04 −	3.294e+04 +	3.309e+04 +	3.009e+04 +	3.006e+04 +	2.893e+04
	std	4.182e+02	4.976e+02	5.226e+02	5.563e+02	2.262e+03	6.473e+03	2.376e+02	7.934e+02	8.016e+02	1.286e+03	6.410e+03	8.901e+03
f_{23}	mean	4.494e+03 +	5.311e+03 +	4.330e+03 +	3.265e+03 +	3.362e+03 +	**2.920e+03** −	7.996e+03 +	3.577e+03 +	3.529e+03 +	4.060e+03 +	2.974e+03 +	2.958e+03
	std	5.959e+01	2.744e+02	7.360e+01	2.932e+01	2.242e+01	1.439e+01	1.508e+02	6.723e+01	5.502e+01	9.046e+01	1.730e+01	1.802e+01
f_{24}	mean	5.297e+03 +	7.603e+03 +	5.126e+03 +	3.768e+03 +	3.890e+03 +	**3.316e+03** −	3.619e+03 +	4.073e+03 +	4.048e+03 +	4.681e+03 +	3.358e+03 ≈	3.359e+03
	std	1.001e+02	6.027e+02	1.401e+02	5.819e+01	3.032e+01	1.056e+01	8.299e+01	8.436e+01	6.431e+01	9.490e+01	2.181e+01	1.715e+01
f_{25}	mean	6.778e+04 +	9.966e+03 +	8.934e+03 +	3.298e+03 −	**3.371e+03** ≈	3.376e+03 ≈	4.022e+03 +	1.197e+04 +	1.291e+04 +	7.733e+03 +	3.383e+03 −	3.388e+03
	std	9.402e+03	1.176e+03	1.170e+03	4.093e+01	3.370e+01	3.879e+01	9.595e+01	1.617e+03	1.895e+03	1.070e+03	3.471e+01	3.157e+01
f_{26}	mean	2.754e+04 +	3.504e+04 +	2.280e+04 +	1.047e+04 +	1.236e+04 +	**6.105e+03** −	7.343e+03 ≈	1.517e+04 +	1.539e+04 +	1.978e+04 +	6.925e+03 +	6.893e+03
	std	1.207e+03	3.096e+03	1.053e+03	4.158e+02	3.046e+02	1.993e+02	1.262e+03	6.947e+02	2.405e+03	9.738e+02	3.190e+02	3.748e+02
f_{27}	mean	4.594e+03 +	7.846e+03 +	3.816e+03 +	3.471e+03 +	3.462e+03 ≈	3.408e+03 −	1.534e+04 +	3.747e+03 +	3.731e+03 +	4.392e+03 +	3.438e+03 +	3.436e+03
	std	2.829e+02	1.066e+03	9.960e+01	3.735e+01	5.467e+01	2.156e+01	8.924e+02	1.164e+02	9.497e+01	1.868e+02	2.603e+01	2.331e+01
f_{28}	mean	2.343e+04 +	1.634e+04 +	1.063e+04 +	**3.411e+03** −	3.476e+03 −	3.550e+03 ≈	4.625e+03 +	1.428e+04 +	1.346e+04 +	1.178e+04 +	3.568e+03 ≈	3.570e+03
	std	1.947e+03	2.187e+03	1.817e+03	3.008e+01	3.322e+01	3.417e+01	1.370e+02	1.977e+03	1.726e+03	1.934e+03	3.432e+01	3.777e+01
f_{29}	mean	2.304e+04 +	1.597e+04 +	1.082e+04 +	6.824e+03 +	6.816e+03 +	**4.501e+03** ≈	7.960e+03 +	8.571e+03 +	8.235e+03 +	1.113e+04 +	4.587e+03 ≈	4.594e+03
	std	2.322e+03	3.249e+03	3.356e+02	3.667e+02	2.897e+02	2.936e+02	7.577e+02	5.134e+02	7.137e+02	8.022e+02	2.241e+02	2.841e+02
f_{30}	mean	3.269e+09 +	2.269e+09 +	8.548e+06 +	**9.972e+03** −	5.413e+04 +	2.659e+04 +	3.687e+05 +	8.903e+07 +	7.789e+07 +	3.232e+07 +	1.764e+04 ≈	1.915e+04
	std	5.006e+08	1.887e+09	1.485e+06	3.556e+03	2.247e+04	8.868e+03	1.445e+05	5.242e+07	5.364e+07	1.197e+07	6.741e+03	5.444e+03
+/≈/−:		29/0/0	29/0/0	26/1/2	13/5/11	13/7/9	6/9/14	21/2/6	25/0/4	24/1/4	27/1/1	14/14/1	-
Avg. ranks:		11.72	10.97	8.24	3.33	4.62	**2.72**	5.83	7.24	7.21	8.45	3.72	3.52

Table 14
Detailed experimental results on 10-D CEC2020.

Func.		DE	PSO	CMA-ES	JADE	L-SHADE	CSO	jSO	MCSO	SCSO	WLCSO	FDBCSO	DFDBCSO
f_1	mean	7.090e+08 +	3.939e+08 +	2.225e+07 +	9.495e+06 +	1.928e+06 +	1.206e+07 +	7.095e+08 +	8.111e+05 +	8.449e+06 +	2.676e+08 +	3.147e+04 +	**2.727e+04**
	std	2.354e+08	4.311e+08	1.022e+07	3.521e+06	6.587e+05	2.785e+06	7.605e+08	4.424e+05	3.868e+06	1.152e+08	1.222e+04	1.139e+04
f_2	mean	2.294e+03 ≈	2.770e+03 +	2.694e+03 +	2.128e+03 −	**1.919e+03** −	2.375e+03 ≈	2.848e+03 +	2.344e+03 ≈	2.537e+03 +	2.202e+03 ≈	2.364e+03 ≈	2.288e+03
	std	2.820e+02	1.742e+02	2.028e+02	1.126e+02	1.905e+02	2.418e+02	8.847e+01	1.418e+02	1.795e+02	2.141e+02	1.936e+02	2.138e+02
f_3	mean	4.265e+04 +	1.567e+03 +	2.585e+03 +	8.207e+02 +	7.929e+02 +	1.052e+03 +	3.323e+03 +	7.806e+02 +	1.143e+03 +	1.162e+04 +	7.524e+02 +	**7.512e+02**
	std	1.074e+04	1.256e+03	6.078e+02	1.922e+01	7.149e+00	7.740e+01	3.314e+03	9.701e+00	1.450e+02	5.212e+03	4.426e+00	6.221e+00
f_4	mean	1.939e+03 +	1.910e+03 +	1.905e+03 +	1.903e+03 +	1.903e+03 ≈	1.903e+03 +	1.906e+03 +	1.903e+03 +	1.904e+03 +	1.907e+03 +	1.903e+03 +	**1.902e+03**
	std	2.819e+01	4.406e+00	7.896e-01	4.993e-01	3.855e-01	5.258e-01	4.369e+00	4.149e-01	7.669e-01	1.493e+00	4.719e-01	5.129e-01
f_5	mean	1.184e+04 ≈	1.124e+05 +	**2.056e+03** −	1.314e+04 ≈	5.597e+03 −	2.147e+04 +	4.273e+03 −	1.083e+04 ≈	9.041e+03 −	2.117e+04 +	1.319e+04 ≈	1.331e+04
	std	3.654e+03	1.360e+05	1.036e+02	4.723e+03	1.637e+03	4.366e+03	3.140e+03	3.511e+03	4.415e+03	9.791e+03	2.957e+03	3.334e+03
f_6	mean	2.890e+03 +	7.328e+03 +	1.973e+03 ≈	3.091e+03 +	2.267e+03 +	2.307e+03 +	1.325e+08 +	**1.772e+03** −	1.853e+03 −	3.318e+03 +	1.905e+03 −	1.988e+03
	std	3.770e+02	4.901e+03	2.186e+02	5.076e+02	1.833e+02	1.399e+02	1.594e+08	1.055e+02	1.516e+02	7.528e+02	1.505e+02	1.742e+02
f_7	mean	5.697e+04 +	9.080e+04 +	6.141e+03 −	2.283e+04 ≈	1.035e+04 −	3.108e+04 +	**4.584e+03** −	2.599e+04 +	1.132e+04 −	5.405e+04 +	1.945e+04 +	1.803e+04
	std	2.127e+04	6.395e+04	1.523e+03	1.162e+04	3.052e+03	1.072e+04	3.016e+03	1.172e+04	4.307e+03	2.874e+04	7.341e+03	5.009e+03
f_8	mean	2.319e+03 +	2.316e+03 +	2.311e+03 +	2.307e+03 +	2.307e+03 +	2.309e+03 +	2.991e+03 +	2.302e+03 +	2.310e+03 +	2.315e+03 +	2.300e+03 ≈	**2.300e+03**
	std	1.041e+00	3.492e+00	1.084e+00	1.167e+00	1.038e+00	1.155e+00	2.376e+02	9.730e-01	9.547e-01	1.407e+00	2.558e-02	2.776e-02
f_9	mean	3.887e+03 +	3.395e+03 +	2.891e+03 +	2.793e+03 +	2.706e+03 +	2.771e+03 +	2.929e+03 +	2.630e+03 +	2.827e+03 +	3.498e+03 +	**2.605e+03** ≈	2.606e+03
	std	1.436e+02	3.402e+02	4.526e+01	1.257e+02	6.289e+01	3.152e+01	3.426e+02	6.612e+00	4.407e+01	2.253e+02	9.843e-01	2.776e+00
f_{10}	mean	3.090e+03 +	3.099e+03 +	2.996e+03 +	2.992e+03 +	2.991e+03 +	2.996e+03 +	3.074e+03 +	2.991e+03 +	2.990e+03 +	3.044e+03 +	2.982e+03 +	**2.980e+03**
	std	3.262e+01	6.832e+01	5.490e+00	5.319e+00	8.185e+00	7.006e+00	3.815e+01	7.418e+00	4.903e+00	2.158e+01	3.640e+00	1.693e+00
+/≈/−:		8/2/0	10/0/0	7/1/2	7/2/1	6/1/3	9/1/0	8/2/0	7/2/1	7/0/3	9/1/0	5/4/1	-
Avg. ranks:		9.90	10.70	6.50	5.90	3.50	7.20	8.90	4.10	5.50	9.20	3.60	**3.00**

Table 15
Detailed experimental results on 20-D CEC2020.

Func.		DE	PSO	CMA-ES	JADE	L-SHADE	CSO	jSO	MCSO	SCSO	WLCSO	FDBCSO	DFDBCSO
f_1	mean	9.612e+09 +	1.120e+09 +	6.605e+08 +	6.721e+06 +	1.544e+05 +	3.157e+06 +	1.109e+08 +	1.415e+06 +	8.544e+07 +	1.528e+09 +	7.480e+02 +	**6.282e+02**
	std	1.597e+09	1.244e+09	9.987e+07	3.093e+06	6.534e+04	7.307e+05	2.291e+08	4.343e+05	7.953e+07	6.436e+08	5.044e+02	2.860e+02
f_2	mean	4.630e+03 −	5.462e+03 +	5.426e+03 +	**2.759e+03** −	2.971e+03 −	4.838e+03 ≈	5.025e+03 ≈	4.870e+03 ≈	5.219e+03 +	4.386e+03 −	4.941e+03 −	4.995e+03
	std	3.396e+02	2.461e+02	2.203e+02	2.620e+02	2.170e+02	3.244e+02	1.101e+02	2.039e+02	2.483e+02	2.463e+02	2.058e+02	2.425e+02
f_3	mean	4.221e+05 +	1.493e+04 +	3.401e+04 +	9.293e+02 +	8.242e+02 ≈	9.815e+02 +	6.096e+03 +	9.107e+02 +	4.164e+03 +	6.403e+04 +	8.191e+02 ≈	**8.178e+02**
	std	6.759e+04	8.037e+03	5.553e+03	3.945e+01	1.011e+01	1.610e+01	7.511e+03	1.712e+01	1.560e+03	2.678e+04	1.065e+01	8.476e+00
f_4	mean	2.461e+04 +	2.339e+03 +	1.947e+03 +	1.911e+03 +	**1.908e+03** −	1.909e+03 +	1.917e+03 +	1.911e+03 +	1.911e+03 +	2.100e+03 +	1.908e+03 ≈	1.909e+03
	std	1.321e+04	9.170e+02	1.107e+01	1.075e+00	1.014e+00	7.347e-01	5.249e+00	1.200e+00	1.251e+00	3.038e+02	8.527e-01	8.181e-01
f_5	mean	1.864e+06 +	8.482e+06 +	4.138e+04 −	3.476e+05 +	1.725e+05 ≈	2.357e+05 ≈	**3.322e+04** −	4.984e+05 +	3.885e+04 −	1.857e+06 +	1.862e+05 ≈	2.005e+05
	std	1.167e+06	6.777e+06	1.063e+04	1.861e+05	6.548e+04	5.465e+04	1.459e+04	2.122e+05	1.442e+04	1.194e+06	9.442e+04	1.080e+05
f_6	mean	5.132e+04 +	1.085e+05 +	3.529e+03 ≈	5.726e+03 +	3.266e+03 ≈	5.342e+03 +	1.842e+06 +	**1.707e+03** −	3.157e+03 ≈	2.117e+04 +	2.968e+03 ≈	3.138e+03
	std	1.660e+04	1.842e+05	5.617e+02	2.595e+03	5.612e+02	9.284e+02	4.543e+06	7.012e+01	8.456e+02	8.040e+03	9.952e+02	9.690e+02
f_7	mean	1.094e+06 +	2.141e+06 +	3.859e+04 +	1.143e+05 +	4.915e+04 +	6.580e+04 +	6.377e+04 ≈	8.923e+04 +	2.493e+04 ≈	1.505e+06 +	1.765e+04 +	**1.726e+04**
	std	7.097e+05	3.164e+06	8.928e+03	5.632e+04	1.660e+04	1.769e+04	2.069e+05	2.289e+04	1.394e+04	6.996e+05	4.116e+03	4.421e+03
f_8	mean	2.399e+03 +	2.398e+03 +	2.345e+03 +	2.325e+03 +	2.312e+03 +	2.320e+03 +	3.766e+03 +	2.304e+03 +	2.335e+03 +	2.364e+03 +	2.300e+03 ≈	**2.300e+03**
	std	9.261e+00	3.765e+01	1.722e+00	2.034e+00	4.955e+00	2.664e+00	2.188e+02	8.213e-01	2.512e+00	8.760e+00	1.783e-01	5.407e-01
f_9	mean	8.958e+03 +	6.193e+03 +	4.561e+03 +	2.719e+03 +	2.614e+03 +	2.742e+03 +	3.170e+03 +	2.677e+03 +	3.515e+03 +	6.258e+03 +	2.601e+03 ≈	**2.601e+03**
	std	4.726e+02	1.736e+03	2.101e+02	5.740e+01	5.270e+00	1.612e+01	4.146e+02	1.320e+01	3.369e+02	5.260e+02	1.389e-01	1.389e-01
f_{10}	mean	3.754e+03 +	4.561e+03 +	3.240e+03 +	3.189e+03 ≈	3.211e+03 +	3.203e+03 ≈	3.600e+03 +	3.208e+03 +	3.153e+03 −	3.457e+03 +	3.207e+03 ≈	3.194e+03
	std	1.655e+02	5.378e+02	3.903e+01	3.541e+01	6.402e+01	4.542e+01	1.892e+02	3.055e+01	2.217e+01	8.296e+01	8.136e+01	5.081e+01
+/≈/−:		9/0/1	10/0/0	8/1/1	9/0/1	5/3/2	8/2/0	7/1/2	8/1/1	7/2/1	9/0/1	2/7/1	-
Avg. ranks:		10.50	10.90	7.70	5.60	3.70	5.60	8.10	5.00	5.60	9.40	**2.90**	3.00

Table 16
Detailed experimental results on 100-D CEC2020.

Func.		DE	PSO	CMA-ES	JADE	L-SHADE	CSO	jSO	MCSO	SCSO	WLCSO	FDBCSO	DFDBCSO
f_1	mean	3.306e+11 +	9.590e+10 +	8.440e+10 +	2.176e+04 +	4.860e+03 ≈	5.640e+03 +	1.301e+09 +	1.546e+10 +	1.173e+11 +	4.164e+10 +	4.627e+03 +	**4.425e+03**
	std	1.994e+10	9.214e+09	1.181e+10	4.795e+04	2.727e+03	2.475e+03	2.840e+08	1.727e+09	1.479e+10	5.400e+09	4.097e+03	5.471e+03
f_2	mean	3.271e+04 +	3.247e+04 +	3.250e+04 +	2.760e+04 −	2.188e+04 −	2.005e+04 −	**1.716e+04** −	3.095e+04 +	3.177e+04 +	2.827e+04 −	3.084e+04 +	3.066e+04
	std	4.777e+02	4.587e+02	3.690e+02	7.420e+02	6.296e+02	2.230e+03	3.254e+02	5.037e+02	4.869e+02	1.021e+03	6.249e+02	3.039e+02
f_3	mean	1.050e+07 +	3.464e+06 +	2.862e+06 +	1.772e+03 +	1.749e+03 +	**1.593e+03** −	4.807e+04 +	5.158e+05 +	3.794e+06 +	1.348e+06 +	1.644e+03 ≈	1.644e+03
	std	7.957e+05	2.757e+05	5.174e+05	1.322e+02	6.280e+01	1.780e+01	1.154e+04	5.470e+04	5.976e+05	2.248e+05	3.249e+01	2.503e+01
f_4	mean	2.043e+07 +	2.751e+05 +	8.304e+04 +	2.023e+03 +	1.987e+03 +	1.952e+03 ≈	2.199e+03 +	3.364e+03 +	1.685e+05 +	3.897e+04 +	1.961e+03 +	**1.950e+03**
	std	7.337e+06	1.515e+05	4.241e+04	2.696e+01	3.976e+00	4.290e+00	7.089e+01	3.697e+02	1.208e+05	1.977e+04	8.946e+00	1.919e+01
f_5	mean	7.317e+08 +	6.200e+08 +	2.740e+07 +	**5.348e+06** −	3.290e+07 +	8.763e+06 +	6.675e+06 +	1.848e+08 +	1.334e+07 +	1.608e+08 +	5.968e+06 +	5.716e+06
	std	1.101e+08	2.733e+08	5.970e+06	1.342e+07	7.893e+06	1.388e+06	1.385e+06	2.973e+07	7.593e+06	8.565e+07	1.297e+06	8.757e+05
f_6	mean	3.248e+09 +	2.868e+08 +	2.597e+05 +	1.601e+05 +	**2.878e+03** −	5.471e+03 ≈	2.587e+04 +	1.201e+04 +	9.499e+06 +	1.508e+06 +	4.778e+03 +	5.188e+03
	std	8.454e+08	4.665e+08	6.721e+04	9.050e+04	3.515e+02	2.067e+03	2.515e+04	2.034e+03	1.115e+07	1.639e+06	1.643e+03	1.958e+03
f_7	mean	4.667e+09 +	4.111e+09 +	3.383e+07 +	2.674e+07 +	6.204e+06 ≈	6.879e+06 +	8.495e+06 +	2.588e+08 +	5.593e+07 +	1.694e+08 +	4.873e+06 +	**4.525e+06**
	std	1.960e+09	3.105e+09	1.398e+07	1.395e+07	3.383e+06	1.459e+06	4.458e+06	3.193e+07	2.769e+07	8.455e+07	1.980e+06	1.638e+06
f_8	mean	4.378e+03 +	9.136e+03 +	3.504e+03 +	2.428e+03 +	2.421e+03 +	**2.315e+03** −	1.458e+04 +	2.660e+03 +	2.912e+03 +	2.879e+03 +	2.374e+03 +	2.366e+03
	std	3.025e+02	1.727e+03	2.057e+02	5.356e+00	7.719e+00	2.064e+01	4.707e+02	1.336e+01	8.552e+01	6.036e+01	3.484e+01	3.190e+01
f_9	mean	1.157e+05 +	1.024e+05 +	6.385e+04 +	2.601e+03 +	2.600e+03 +	2.608e+03 +	1.835e+04 +	2.605e+04 +	1.233e+05 +	4.165e+04 +	2.600e+03 ≈	**2.600e+03**
	std	1.195e+04	8.496e+03	8.921e+03	1.850e+00	2.034e-02	5.374e-01	3.204e+03	7.327e+02	8.882e+03	3.123e+03	2.409e-05	2.417e-05
f_{10}	mean	4.251e+04 +	1.201e+04 +	8.541e+03 +	3.700e+03 +	**3.504e+03** −	3.536e+03 −	4.401e+03 +	4.760e+03 +	1.004e+04 +	6.757e+03 +	3.539e+03 +	3.583e+03
	std	6.442e+03	1.167e+03	9.039e+02	1.129e+02	2.761e+01	5.001e+01	9.844e+01	1.150e+02	1.107e+03	5.008e+02	5.042e+01	4.901e+01
+/≈/−:		10/0/0	10/0/0	10/0/0	8/0/2	5/2/3	4/2/4	9/0/1	10/0/0	10/0/0	9/0/1	8/2/0	-
Avg. ranks:		11.70	10.60	8.70	4.70	3.40	3.00	5.80	7.40	9.50	7.90	2.90	**2.40**

Table 17
Detailed experimental results on 10-D CEC2022.

Func.		DE	PSO	CMA-ES	JADE	L-SHADE	CSO	jSO	MCSO	SCSO	WLCSO	FDBCSO	DFDBCSO
f_1	mean	6.871e+03 +	2.083e+03 +	4.305e+02 +	1.730e+03 +	4.844e+02 +	4.916e+02 +	5.675e+02 +	3.253e+02 +	3.376e+02 +	3.108e+03 +	3.003e+02 +	**3.003e+02**
	std	1.604e+03	1.649e+03	4.487e+01	7.868e+02	7.272e+01	5.222e+01	2.812e+02	6.670e+00	1.778e+01	9.371e+02	1.872e-01	1.030e-01
f_2	mean	4.545e+02 +	4.901e+02 +	4.145e+02 +	4.172e+02 +	4.098e+02 +	4.278e+02 +	4.167e+02 ≈	**4.009e+02** ≈	4.111e+02 +	4.286e+02 ≈	4.054e+02 ≈	4.069e+02
	std	1.383e+01	3.474e+01	2.267e+00	5.368e+00	6.641e+00	9.905e+00	2.709e+01	1.191e+00	3.647e+00	1.223e+01	1.390e+01	1.394e+01
f_3	mean	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	**6.000e+02** ≈	6.000e+02
	std	1.503e-02	6.061e-03	9.061e-04	1.758e-05	5.979e-06	9.825e-05	2.701e-02	7.600e-06	1.778e-04	6.174e-03	9.620e-08	1.702e-07
f_4	mean	9.174e+02 +	8.850e+02 +	8.778e+02 +	8.573e+02 +	8.491e+02 +	8.384e+02 ≈	8.584e+02 +	8.376e+02 ≈	8.499e+02 +	8.633e+02 +	**8.371e+02** ≈	8.375e+02
	std	1.574e+01	1.785e+01	1.143e+01	1.500e+01	1.298e+01	1.031e+01	2.577e+00	7.692e+00	1.657e+01	1.141e+01	8.714e+00	8.317e+00
f_5	mean	9.028e+02 +	9.011e+02 +	9.005e+02 +	9.003e+02 +	9.002e+02 +	9.000e+02 +	9.009e+02 +	9.000e+02 +	9.001e+02 +	9.008e+02 +	**9.000e+02** ≈	9.000e+02
	std	6.007e-01	7.271e-01	1.301e-01	9.377e-02	7.531e-02	1.090e-02	3.999e-01	4.825e-04	3.220e-02	3.118e-01	1.395e-05	1.389e-05
f_6	mean	1.059e+05 +	1.530e+06 +	**4.214e+03** −	5.864e+04 +	1.988e+04 ≈	5.897e+04 +	7.699e+05 +	3.603e+04 +	3.041e+04 +	5.765e+04 +	1.784e+04 ≈	1.914e+04
	std	3.774e+04	3.350e+06	7.622e+02	2.726e+04	8.658e+03	2.059e+04	2.667e+06	1.354e+04	8.778e+03	1.910e+04	4.201e+03	5.474e+03
f_7	mean	2.068e+03 +	2.091e+03 +	2.077e+03 +	2.035e+03 +	2.034e+03 +	2.046e+03 +	2.035e+03 ≈	2.041e+03 +	2.058e+03 +	2.081e+03 +	2.033e+03 +	**2.032e+03**
	std	1.579e+01	3.645e+01	1.255e+01	3.579e+00	3.036e+00	3.942e+00	1.328e+01	5.639e+00	9.533e+00	1.674e+01	1.828e+00	2.471e+00
f_8	mean	2.255e+03 +	3.167e+03 +	**2.229e+03** −	2.251e+03 +	2.231e+03 −	2.280e+03 +	7.189e+08 ≈	2.232e+03 −	2.234e+03 ≈	2.283e+03 +	2.237e+03 ≈	2.238e+03
	std	8.929e+00	1.179e+03	1.529e+00	1.295e+01	2.189e+00	5.547e+01	2.802e+09	3.196e+00	2.648e+00	3.031e+01	1.122e+01	9.729e+00
f_9	mean	2.668e+03 +	2.634e+03 +	2.525e+03 +	2.477e+03 +	2.319e+03 +	2.317e+03 +	2.566e+03 +	2.301e+03 +	2.332e+03 +	2.536e+03 +	2.300e+03 +	**2.300e+03**
	std	1.826e+01	1.429e+02	1.507e+02	9.423e+01	1.275e+01	1.080e+01	2.140e+02	2.177e-01	7.585e+01	1.033e+02	8.280e-02	9.467e-02
f_{10}	mean	2.647e+03 +	2.643e+03 +	2.629e+03 +	2.636e+03 +	2.632e+03 +	2.639e+03 +	3.535e+03 +	2.628e+03 +	2.628e+03 +	2.635e+03 +	2.642e+03 +	**2.626e+03**
	std	4.901e+00	1.182e+01	1.266e+00	4.995e+00	3.344e+00	2.203e+01	5.396e+02	2.507e+01	1.134e+00	3.675e+00	3.556e+01	1.096e+01
f_{11}	mean	2.675e+03 +	2.635e+03 +	2.615e+03 +	2.606e+03 +	2.601e+03 +	2.604e+03 +	2.614e+03 +	2.600e+03 +	2.606e+03 +	2.622e+03 +	2.600e+03 +	**2.600e+03**
	std	1.839e+02	2.402e+01	3.900e+01	4.297e+00	4.151e-01	9.735e-01	1.137e+01	4.316e-02	4.030e+00	6.886e+00	1.630e-02	1.163e-02
f_{12}	mean	2.868e+03 +	2.901e+03 +	2.867e+03 +	2.867e+03 +	2.867e+03 +	2.869e+03 +	4.170e+03 +	2.877e+03 +	2.867e+03 +	2.870e+03 +	2.867e+03 +	**2.867e+03**
	std	5.400e-01	2.488e+01	6.736e-01	4.992e-01	6.631e-01	6.642e-01	3.365e+02	1.942e+00	3.799e-01	8.067e-01	2.270e-01	3.204e-01
+/≈/−:		12/0/0	12/0/0	10/0/2	12/0/0	10/1/1	11/1/0	9/3/0	9/2/1	11/1/0	11/1/0	6/6/0	-
Avg. ranks:		10.67	10.92	6.17	6.67	4.17	6.67	9.50	4.00	5.42	9.25	2.58	**2.00**

Table 18
Detailed experimental results on 20-D CEC2022.

Func.		DE	PSO	CMA-ES	JADE	L-SHADE	CSO	jSO	MCSO	SCSO	WLCSO	FDBCSO	DFDBCSO
f_1	mean	3.345e+04 +	1.247e+04 +	2.599e+03 +	7.306e+03 +	5.799e+02 +	6.071e+02 +	5.823e+02 +	1.253e+03 +	7.738e+02 +	1.506e+04 +	3.000e+02 ≈	**3.000e+02**
	std	3.834e+03	6.543e+03	5.082e+02	1.770e+03	1.007e+02	1.061e+02	3.768e+02	2.135e+02	2.932e+02	4.345e+03	2.683e-02	1.560e-02
f_2	mean	1.076e+03 +	8.834e+02 +	5.142e+02 +	4.642e+02 +	4.553e+02 +	4.527e+02 +	5.203e+02 +	4.602e+02 +	4.603e+02 +	5.809e+02 +	**4.496e+02** ≈	4.500e+02
	std	1.132e+02	4.003e+02	1.278e+01	9.287e+00	9.406e+00	2.731e+00	4.775e+01	3.284e+00	1.239e+01	3.920e+01	7.664e-01	1.082e+00
f_3	mean	6.004e+02 +	6.001e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.000e+02 +	6.001e+02 +	6.000e+02 +	**6.000e+02**
	std	6.551e-02	7.746e-02	5.129e-03	4.557e-05	4.372e-07	2.739e-05	5.950e-03	1.493e-05	1.697e-03	2.183e-02	1.106e-09	1.812e-09
f_4	mean	1.153e+03 +	1.110e+03 +	1.070e+03 +	1.015e+03 +	9.819e+02 +	9.632e+02 ≈	9.687e+02 ≈	9.689e+02 ≈	9.758e+02 ≈	1.039e+03 +	**9.602e+02** ≈	9.662e+02
	std	3.906e+01	3.032e+01	2.553e+01	3.073e+01	2.389e+01	2.341e+01	1.355e+01	1.596e+01	2.949e+01	3.231e+01	2.458e+01	1.675e+01
f_5	mean	9.156e+02 +	9.038e+02 +	9.036e+02 +	9.022e+02 +	9.009e+02 +	9.000e+02 +	9.021e+02 +	9.000e+02 +	9.002e+02 +	9.033e+02 +	**9.000e+02** ≈	9.000e+02
	std	2.647e+00	1.505e+00	5.355e-01	6.104e-01	2.976e-01	1.566e-03	2.836e-01	1.364e-03	1.953e-01	1.062e+00	5.609e-06	2.456e-05
f_6	mean	2.174e+08 +	3.198e+08 +	4.592e+06 +	8.157e+06 +	1.885e+06 +	2.189e+06 +	**2.559e+04** ≈	9.386e+06 +	4.969e+04 ≈	3.357e+07 +	2.392e+05 ≈	2.709e+05
	std	8.385e+07	3.004e+08	1.363e+06	4.864e+06	8.135e+05	8.849e+05	4.538e+03	3.117e+06	1.069e+04	1.755e+07	8.537e+04	9.382e+04
f_7	mean	2.907e+03 +	3.067e+03 +	2.276e+03 +	2.135e+03 +	2.102e+03 +	2.051e+03 +	2.273e+03 +	2.068e+03 +	2.054e+03 +	2.592e+03 +	2.040e+03 +	**2.039e+03**
	std	2.108e+02	4.558e+02	6.559e+01	1.864e+01	2.117e+01	4.363e+00	3.324e+02	1.220e+01	7.715e+00	1.943e+02	2.557e+00	3.855e+00
f_8	mean	3.497e+04 +	7.386e+07 +	2.434e+03 −	3.489e+03 ≈	2.693e+03 −	3.897e+03 ≈	**2.389e+03** −	4.413e+03 +	2.418e+03 −	5.692e+03 +	3.640e+03 ≈	3.701e+03
	std	3.174e+04	1.672e+08	7.161e+01	5.184e+02	2.137e+02	3.922e+02	1.728e+02	1.168e+03	2.620e+02	2.001e+03	4.276e+02	3.995e+02
f_9	mean	2.782e+03 +	3.166e+03 +	2.650e+03 ≈	2.676e+03 +	2.658e+03 +	2.665e+03 +	**2.619e+03** −	2.678e+03 +	2.647e+03 ≈	2.749e+03 +	2.650e+03 ≈	2.650e+03
	std	3.651e+01	1.688e+02	6.247e+00	1.009e+01	5.267e+00	3.688e+00	1.280e+02	5.032e+00	5.592e+00	3.899e+01	3.956e+00	4.921e+00
f_{10}	mean	3.851e+03 +	2.967e+03 +	3.468e+03 +	3.030e+03 +	2.844e+03 +	2.823e+03 +	5.692e+03 +	2.855e+03 +	2.830e+03 +	2.859e+03 +	**2.810e+03** ≈	2.821e+03
	std	1.357e+03	2.445e+02	1.314e+03	6.177e+02	1.186e+01	4.794e+01	2.108e+02	7.832e+01	4.241e+01	1.602e+01	2.592e+01	3.168e+01
f_{11}	mean	2.738e+03 +	2.720e+03 +	2.620e+03 +	2.605e+03 +	2.600e+03 +	2.601e+03 +	2.606e+03 +	2.600e+03 +	2.612e+03 +	2.639e+03 +	2.600e+03 ≈	**2.600e+03**
	std	5.633e+01	8.153e+01	2.316e+00	2.715e+00	6.495e-02	3.666e-01	6.645e+00	5.781e-02	3.409e+00	6.737e+00	2.860e-03	2.076e-03
f_{12}	mean	2.955e+03 +	3.239e+03 +	2.955e+03 +	2.950e+03 +	2.949e+03 +	2.957e+03 +	5.181e+03 +	2.995e+03 +	2.956e+03 +	2.988e+03 +	**2.945e+03** ≈	2.945e+03
	std	4.081e+00	9.708e+01	5.264e+00	2.927e+00	2.429e+00	9.369e+00	3.726e+01	5.246e+00	1.466e+01	1.168e+01	3.531e+00	4.879e+00
+/≈/−:		12/0/0	12/0/0	10/1/1	11/1/0	11/0/1	10/2/0	8/2/2	11/1/0	8/3/1	12/0/0	2/10/0	-
Avg. ranks:		11.08	11.00	7.58	7.00	4.58	5.00	6.17	6.33	4.92	9.58	**2.25**	2.50