Accelerating differential evolution algorithm with Gaussian sampling based on estimating the convergence points

1st Rui Zhong
Graduate School of Information Science and Technology
Hokkaido University
Sapporo, Japan
rui.zhong.u5@elms.hokudai.ac.jp

2nd Masaharu Munetomo
Information Initiative Center
Hokkaido University
Sapporo, Japan
munetomo@iic.hokudai.ac.jp

***Abstract*—In this paper, we propose a simple strategy for estimating the convergence point approximately by averaging the elite sub-population. Based on this idea, we derive two methods, which are ordinary averaging strategy, and weighted averaging strategy. We also design a Gaussian sampling operator with the mean of the estimated convergence point with a certain standard deviation. This operator is combined with the traditional differential evolution algorithm (DE) to accelerate the convergence. Numerical experiments show that our proposal can accelerate the DE on most functions of 28 low-dimensional test functions on the CEC2013 Suite, and our proposal can easily be extended to combine with other population-based evolutionary algorithms with a simple modification.**

***Index Terms*—Estimation of convergence point, Gaussian sampling, Acceleration, Averaging strategy**

I. Introduction

As an important part of artificial intelligence, evolutionary computation has achieved great success in solving continuous [1], large-scale [2], constraint [3], and multi-objective optimization problems [4] in the past few decades. However, according to the No Free Lunch Theorem (NFLT) [5], there is no optimization algorithm that can solve all optimization problems perfectly. NFLT proves that the average performance of any pair of algorithms A and B is identical on all possible problems. Therefore, if an algorithm performs well on a certain class of problems, it must pay for that with performance degradation on the remaining problems, since this is the only way for all algorithms to have the same performance on average across all functions. This hypothesis has a great impact on the scientific community for the notion that there is no universal optimizer. In addition, a single algorithm often cannot balance the exploration and the exploitation well. For several reasons, hybrid algorithms or memetic algorithms appears, the hybrid algorithm was first proposed to solve the Traveling Salesman Problem (TSP) by modifying the genetic algorithm with a local search operator [6]. However, NFLT also limits the hybrid algorithms with identical average performance on all possible problems, thus the features of the problem become the starting point for designing a suitable optimization algorithm.

Since hybrid algorithms were not proposed as specific optimization algorithms, but as a broad class of algorithms inspired by the diffusion of ideas and composed of multiple existing operators, the community started showing increasing attention toward these algorithmic structures as a general guideline for addressing specific problems.

In addition, many studies [7], [8] show that it is a promising strategy to find trusting regions or elites in the fitness landscape, which can be fully applied to guide the direction of evolution well. Murata et al [9]. first proposed that a mathematical method could be used to calculate the global optimum using the information of two subsequent generations. Yu et al [10]. proposed that in the differential evolution algorithm, the differential vector from the parent individual to its offspring (or the vector from an individual with poor fitness to an individual with higher fitness) is defined as the moving vector, and the convergence point is estimated according to the moving vector.

In this paper, we employ a simple strategy to approximately estimate the convergence point, which is roughly estimated by the averaging strategy and weighted averaging strategy of the elite sub-population. Besides, after each generation optimized by DE, we apply a Gaussian sampling operator with the mean of the estimated convergence point. In experiment design, we evaluate our proposal with 28 benchmark functions from the CEC 2013 Suite [11]. Finally, we summarize our work and provide some open topics for future discussions.

The rest of the paper is organized as follows. Section II includes preliminaries and related works. Section III provides a detailed description of our proposal. Section IV provides the numerical experiments and experiments analysis. Finally, Section V concludes the paper and shows the future directions.

II. Preliminaries and Related works

In this section, we first introduce the preliminaries of this work, including the Random Search (RS), Genetic Algorithm (GA), Differential Evolution Algorithm (DE), Evolution Strategy (ES), and Particle Swarm Optimization (PSO). Finally, we introduce the original estimation of a convergence point.

A. Preliminaries

1) RS: RS was proposed by Rastrigin in 1963, and an early introduction to RS with basic mathematical analysis was given in the paper [12]. The principle of RS is to iteratively move to better positions in the search space, these positions are randomly sampled around the current optimum. RS is an optimization method that does not require the gradients of the fitness landscape, so RS can be employed for discrete or differentiable problems. The Pseudocode of RS is shown in Algorithm 1.

Algorithm 1: RS

Input: Dimension : D; Search space : S; Generation : T
Output: Best solution : E

Function `RS` (D, S, T) **:**
- $t \leftarrow 0$
- ▶ (Solution initialization)
- $P_t \leftarrow$ **Initial**(D, S)
- $FP \leftarrow$ **Evaluate**(P_t)
- $E \leftarrow P_t$
- **while** $t < T$ and not convergence **do**
  - $O_t \leftarrow$ **Sampling**(P_t)
  - $FO \leftarrow$ **Evaluate**(O_t)
  - **if** $FO > FP$ **then**
    - $P_t \leftarrow O_t$
    - $E \leftarrow O_t$
  - **end**
  - $t \leftarrow t + 1$
- **end**
- **return** E

2) GA: GA [13] simulates the biological evolution process of chromosomes with selection, crossover, and mutation. Chromosomes present problem solutions and are evaluated based on the fitness function to select parents. Selection is an important process for choosing parents to produce the offspring and can affect the convergence of GA, Mutation increases the diversity of the population by randomly modifying the genes in the chromosome with a certain probability. The Pseudocode of GA is shown in Algorithm 2

Algorithm 2: GA

Input: Dimension : D; Search space : S; Population size : PS; Generation : T
Output: Best solution : E

Function `GA` (D, S, PS, T) **:**
- $t \leftarrow 0$
- ▶ (Population initialization)
- $P_t \leftarrow$ **Initial**(D, S, PS)
- $F_t \leftarrow$ **Evaluate**(P_t)
- $E \leftarrow$ **bestIndividual**(P_t, F_t)
- **while** $t < T$ and not convergence **do**
  - ▶ (Selection, Crossover and Mutation)
  - $O_t \leftarrow$ **Selection**(P_t, F_t)
  - $O_t \leftarrow$ **Crossover**(O_t)
  - $P_{t+1} \leftarrow$ **Mutation**(O_t)
  - $F_{t+1} \leftarrow$ **Evaluate**(P_{t+1})
  - $E \leftarrow$ **bestIndividual**(P_{t+1}, F_{t+1})
  - $t \leftarrow t + 1$
- **end**
- **return** E

3) DE: DE was first proposed by Storn and Price in 1995 [14]. DE has been applied to solve many complex optimization problems, and the procedure is similar to the GA, DE mainly includes mutation, crossover, and selection. The Pseudocode of DE is shown in Algorithm 3.

Algorithm 3: DE

Input: Dimension : D; Search space : S; Population size : PS; Generation : T
Output: Best solution : E

Function `DE` (D, S, PS, T) **:**
- $t \leftarrow 0$
- ▶ (Population initialization)
- $P_t \leftarrow$ **Initial**(D, S, PS)
- $F_t \leftarrow$ **Evaluate**(P_t)
- $E \leftarrow$ **bestIndividual**(P_t, F_t)
- **while** $t < T$ and not convergence **do**
  - ▶ (Selection, Crossover and Mutation)
  - $O_t \leftarrow$ **Mutation**(P_t, F_t)
  - $O_t \leftarrow$ **Crossover**(O_t)
  - $F_{t+1} \leftarrow$ **Evaluate**(O_t)
  - $P_{t+1} \leftarrow$ **Selection**(O_t, F_{t+1})
  - $E \leftarrow$ **bestIndividual**(P_{t+1}, F_{t+1})
  - $t \leftarrow t + 1$
- **end**
- **return** E

Mutation is an essential process in DE. We briefly introduce 2 common mutation strategies:

$$\textbf{DE/rand/1}: V_i(g) = X_{p1}(g) + F \cdot (X_{p2}(g) - X_{p3}(g))$$
$$\textbf{DE/best/1}: V_i(g) = X_{best}(g) + F \cdot (X_{p1}(g) - X_{p2}(g))$$

$X_{pi}(g)$ is the different individuals randomly selected from the current population, $X_{best}(g)$ is the best individual in the current population. F is the scaling factor. The differential vectors between individuals are calculated in the mutation is the origin of the name of DE.

4) ES: ES was proposed in 1963 [15], as an optimization algorithm, ES imitates the mechanism of biological evolution. Under the hypothesis that no matter what changes occur in genes, the results or traits always follow a Gaussian distribution with zero mean and a certain variance. Although ES employs mutation and crossover to generate offspring which is similar to GA, ES emphasizes the phenotype rather than the genotype, and ES also applies the real coding instead of binary coding in GA. The Pseudocode of ES is shown in Algorithm

4.

Algorithm 4: ES

Input: Dimension : D; Search space : S; Population size : PS; Generation : T
Output: Best solution : E
Function **ES** (D, S, PS, T) **:**
  $t \leftarrow 0$
  ▶ (Population initialization)
  $P_t \leftarrow$ **Initial**(D, S, PS)
  $FP_t \leftarrow$ **Evaluate**(P_t)
  $E \leftarrow$ **bestIndividual**(P_t, F_t)
  while $t < T$ and not convergence **do**
    ▶ (Selection, Crossover and Mutation)
    $O_t \leftarrow$ **Selection**(P_t)
    $O_t \leftarrow$ **Crossover**(O_t)
    $O_t \leftarrow$ **Mutation**(O_t)
    $FO_t \leftarrow$ **Evaluate**(O_t)
    $P_{t+1}, FP_{t+1} \leftarrow$ **Offspring**(O_t, FO_t, P_t, FP_t)
    $E \leftarrow$ **bestIndividual**(P_{t+1}, FP_{t+1})
    $t \leftarrow t + 1$
  end
  return E

5) PSO: PSO simulates the behavior of bird flocking, fish schooling, and swarming theory to realize the optimization process [16]. The original version of PSO as developed by the authors comprises a very simple concept, and paradigms can be implemented in a few lines of computer code. It requires only primitive mathematical operators and is computationally inexpensive in terms of both memory requirements and speed. The Pseudocode of PSO is shown in Algorithm 5. We update the position and velocity by Eq (1) in lines 13 and 14 of the Algorithm 5

$$\begin{aligned} x_{id}^{k+1} &= x_{id}^{k} + v_{id}^{k+1} \\ v_{id}^{k+1} &= w v_{id}^{k} + c_1 r_1 (p_{id,pbest}^{k} - x_{id}^{k}) + \\ &\quad c_2 r_2 (p_{d,gbest}^{k} - x_{id}^{k}) \end{aligned} \tag{1}$$

i denotes the i^{th} individual, d denotes in the d^{th} dimension, k denotes in the k^{th} generation. In the velocity update equation, w is a inertia weight, c_1 and c_2 are learning factor, r_1 and r_2 are random number.

Algorithm 5: PSO

Input: Dimension : D; Search space : S; Population size : PS; Generation : T
Output: Best solution : $gbest$
1 **Function** **PSO** (D, S, PS, T) **:**
2   $t \leftarrow 0$
3   ▶ (Initialize the position and velocity)
4   $X_t, V_t \leftarrow$ **Initial**(D, S, PS)
5   **for** $i = 0$ *to* PS **do**
6     $pbest_i \leftarrow X_{t,i}$
7   **end**
8   $F_t \leftarrow$ **Evaluate**(X_t)
9   $gbest \leftarrow$ **bestIndividual**(X_t, F_t)
10   **while** $t < T$ and not convergence **do**
11   ▶ (Optimization)
12   **for** $i = 0$ *to* PS **do**
13       $X_{t+1,i} \leftarrow$ **Update**$(X_{t,i}, V_{t,i})$
14       $V_{t+1,i} \leftarrow$ **Update**$(V_{t,i}, pbest_i, gbest)$
15   **end**
16     $F_{t+1} \leftarrow$ **Evaluate**(X_{t+1})
17   **for** $i = 0$ *to* PS **do**
18   **if** $F_{t+1,i} > F_{pbest,i}$ **then**
19         $pbest_i \leftarrow X_{t+1,i}$
20   **end**
21   **if** $F_{pbest,i} > F_{gbest,i}$ **then**
22         $gbest_i \leftarrow pbest_i$
23   **end**
24   **end**
25     $t \leftarrow t + 1$
26   **end**
27   **return** $gbest$

B. Estimation of a convergence point

The concept of estimation of a convergence point was first proposed by Murata. The paper [9] hypothesize that: in a population-based optimization algorithm, all individuals are moving towards the global optimal solution. According to this hypothesis, the nearest point to the extensions of all their movement directions should locate near the global optimal point. Fig.1 shows how the estimation of a convergence point works. But in practice, the estimated point is not exactly on the global optimum due to incorrect directions or inaccurate directions of population movements. However, it is highly expected that the estimation point is close to the global optimum at least for unimodal functions.

Let us derive how to estimate the convergence point mathematically. First, we define parent (worse) individual $\boldsymbol{p}_i$, offspring (better) $\boldsymbol{o}_i$, and moving vector $\boldsymbol{d}_i$ as describe in Fig 1. The unit direction vector of $\boldsymbol{d}_i$ is given as $\boldsymbol{d}_{0i} = \frac{\boldsymbol{d}_i}{||\boldsymbol{d}_i||}$,i.e., $\boldsymbol{d}_{0i}^{\mathrm{T}} \boldsymbol{d}_{0i} = 1$. $\mathbf{X}$ denotes the estimated convergence point, and $\boldsymbol{p}_i + t_i \boldsymbol{d}_i$ represents the expansion from parent individual $\boldsymbol{p}_i$ with the direction $\boldsymbol{d}_i$. $L(\mathbf{X}, t_i)$ in Eq (2) becomes the minimum.

$$\min(L(\mathbf{X}, t_i)) = \min(\sum_{i=1}^{n} ||\boldsymbol{p}_i + t_i \boldsymbol{d}_i - \mathbf{X}||^2) \tag{2}$$

As the minimum line segment from the convergence point $\mathbf{X}$ to the expansion line segments is the orthogonal projection from $\mathbf{X}$, we can apply the Eq (3) into Eq (2) to remote t_i.

$$\boldsymbol{d}_i^T (\boldsymbol{p}_i + t_i \boldsymbol{d}_i - \mathbf{X}) = 0 \text{ (orthogonal condition)} \tag{3}$$

Finally, the convergence point $\mathbf{X}$ can be calculated by Eq (4). See detail expansion of equations in paper [9].

$$\widehat{\mathbf{X}} = \left\{ \sum_{i=1}^{n} (\boldsymbol{I}_d - \boldsymbol{d}_{0i} \boldsymbol{d}_{0i}^{\mathrm{T}}) \right\}^{-1} \left\{ \sum_{i=1}^{n} (\boldsymbol{I}_d - \boldsymbol{d}_{0i} \boldsymbol{d}_{0i}^{\mathrm{T}}) \boldsymbol{p}_i \right\} \tag{4}$$

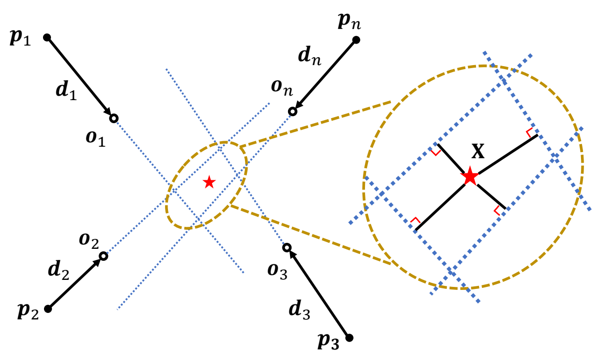
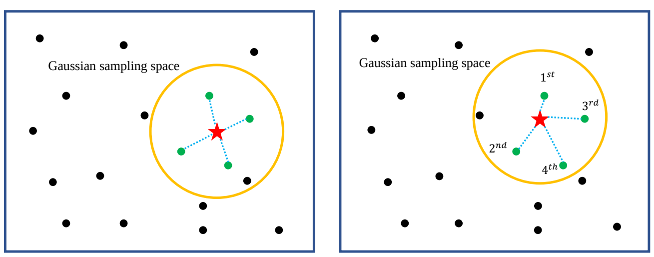


Fig. 1. Moving vector $\boldsymbol{d}_i (= \boldsymbol{o}_i - \boldsymbol{p}_i)$ is calculated from a parent (worse) individual $\boldsymbol{p}_i$ and its offspring (better) $\boldsymbol{o}_i$. The $\star$ is the estimated convergence point.

III. Proposal

In this Section, we will introduce our proposal in detail. First, we propose two approximate methods for estimating the convergence points with Gaussian sampling, which are shown in Fig 2



(a) (b)

Fig. 2. (a).elite sub-population averaging strategy for approximating the convergence point. (b).elite sub-population weighted averaging strategy for approximating the convergence point. The orange circle is the Gaussian sampling space with the mean of estimated convergence point.

First, we proposed a hypothesis that there is a higher probability for a optimum exists around an elite sub-population (individuals with higher fitness). Based on this hypothesis, we select the elite sub-population from the individual population of each generation from DE, and apply the averaging strategy and the weighted averaging strategy to approximately estimate the convergence point. Algorithm 6 and Algorithm 7 show the procedure of averaging strategy and weighted averaging strategy.

In Algorithm 7, we calculate the weights based on normalized fitness of individuals. After the convergence point is estimated, we apply a Gaussian sampling operator with the mean of the convergence point and a certain standard deviation σ to find k individuals. Then we select the best k individuals among 1 estimated convergence point and k individuals found by Gaussian sampling operator and worst

Algorithm 6: Averaging strategy

Input: Population : P; Elite rate : r; Dimension : D
Output: Estimated convergence point : C

Function `AS` (P, r) **:**
  $EP \leftarrow$ **bestIndividuals**(P, r)
  $s \leftarrow$ **size**(EP)
  for $i = 0$ *to* D **do**
    $C_i \leftarrow 0$
    for $j = 0$ *to* s **do**
      $C_i \leftarrow C_i + EP_{i,j}$
    end
    $C_i \leftarrow C_i / s$
  end
  return C

Algorithm 7: Weighted averaging strategy

Input: Population : P; Elite rate : r; Dimension : D
Output: Estimated convergence point : C

Function `WAS` (P, r) **:**
  $EP \leftarrow$ **bestIndividuals**(P, r)
  $EF \leftarrow$ **Evaluate**(EP)
  $s \leftarrow$ **size**(EP)
  $F \leftarrow 0$
  for $i = 0$ *to* s **do**
    $F \leftarrow F + EF_i$
  end
  ▶ (Calculate the weights)
  for $i = 0$ *to* s **do**
    $w_i \leftarrow EF_i / F$
  end
  for $i = 0$ *to* D **do**
    $C_i \leftarrow 0$
    for $j = 0$ *to* s **do**
      $C_i \leftarrow C_i + w_j * EP_{i,j}$
    end
  end
  return C

k individuals generated by DE as a part of offspring to participate the optimization.

IV. Numerical Experiment

Here, we define shortened names of our proposal:

(a) P1: proposal with averaging strategy and Gaussian sampling operator.

(b) P2: proposal with weighted averaging strategy and Gaussian sampling operator.

We apply 28 benchmark functions from the CEC2013 Suite [11] with independent 30 trial runs in this evaluation experiment. Table I shows the details of these functions. And we compare our proposal with RS, GA, DE, ES and PSO. The parameter of these algorithm is shown in Table II

At the end of optimization, we apply the Kruskal-Wallis test and the Holm multiple comparison test on the fitness

TABLE I
CEC2013 SUITE: UNI=UNIMODAL, MULTI=MULTIMODAL, COMP=COMPOSITION

Func.	Types	Characteristics	Optimum
f_1	Uni	Sphere function	-1400
f_2		Rotated high conditioned elliptic function	-1300
f_3		Rotated Bent Cigar function	-1200
f_4		Rotated discus function	-1100
f_5		Different powers function	-1000
f_6	Multi	Rotated Rosenbrock ’ s function	-900
f_7		Rotated Schaffers function	-800
f_8		Rotated Ackley ’ s function	-700
f_9		Rotated Weierstrass function	-600
f_{10}		Rotated Griewank ’ s function	-500
f_{11}		Rastrigin ’ s function	-400
f_{12}		Rotated Rastrigin ’ s function	-300
f_{13}		Non-continuous rotated Rastrigin ’ s function	-200
f_{14}		Schwefel ’ s function	-100
f_{15}		Rotated Schwefel ’ s function	100
f_{16}		Rotated Katsuura function	200
f_{17}		Lunacek bi-Rastrigin function	300
f_{18}		Rotated Lunacek bi-Rastrigin function	400
f_{19}		Expanded Griewank ’ s plus Rosenbrock ’ s function	500
f_{20}		Expanded Schaffer ’ s f_6 function	600
f_{21}	Comp	Composition function 1 (n = 5, rotated)	700
f_{22}		Composition function 2 (n = 3, unrotated)	800
f_{23}		Composition function 3 (n = 3, rotated)	900
f_{24}		Composition function 4 (n = 3, rotated)	1000
f_{25}		Composition function 5 (n = 3, rotated)	1100
f_{26}		Composition function 6 (n = 5, rotated)	1200
f_{27}		Composition function 7 (n = 5, rotated)	1300
f_{28}		Composition function 8 (n = 5, rotated)	1400

TABLE II
THE PARAMETER OF COMPARISON METHODS

Alg.	Parameter	Value
Common parameters	dimension	2-D, 10-D, 30-D
	population size	50 * dimension
	max evaluation times	1000 * dimension
	search space	[-100, 100] * dimension
RS	sampling strategy	random sampling
GA	strategy	Elitist GA
	crossover rate	0.5
	mutation rate	0.1
DE	strategy	DE/current-to-best/1
	scale factor	0.7
	crossover rate	0.9
ES	strategy	(1+1)-ES
	mutation strength	0.2
PSO	w	0.9
	c_1, c_2	2
Proposal	strategy	DE/current-to-best/1
	scale factor	0.7
	crossover rate	0.9
	elite proportion	0.05
	σ in Gaussian sampling	5

values. Table III shows their results of the statistical tests. In Holm multiple comparison test, if our proposal is significant better than the second-best algorithm or original DE, we apply $\gg$ (p<0.01) and $>$ (p<0.05) to denote the significance, $\approx$ represents there is no significance between two comparing methods.

From the experiment results, we can see that our proposal can significantly accelerate the DE in most test functions on the CEC2013 Suite, and P1 has no siginificant difference with P2 in most functions. In the 2-D space, our proposal is competitive with the comparing 5 methods, while in the 10-D and 50-D space, although PSO outperforms our proposal combined with DE on most test functions, this is mainly due to the limited performance of DE on these functions, our proposal can still accelerate the DE in optimization. Different from the method proposed by Yu et al [10], it is unnecessary for our proposal to calculate the moving vector, so it has stronger scalability for our proposal combined with other evolutionary algorithms.

V. CONCLUSION

In this paper, we propose a simple strategy to estimate the convergence point approximately. Based on the averaging strategy, ordinary averaging strategy and weighted averaging strategy are derived. We also design a Gaussian sampling operator based on the estimated convergence point and combine it with traditional DE. Numerical experiments show our proposal can improve the performance of traditional DE. Although our proposal combined with DE is worse than PSO in some functions with 10-D and 50-D, the main reason is the performance of DE is limited in these functions, and our proposal has strong scalability that can be extended to all population-based heuristic algorithms in theory.

In future research, a well-performed local search operator replacing the Gaussian sampling is a promising topic, such as the introduction of CMA-ES or the adaptive evolution strategy. And in this paper, we only apply our proposal in low dimensional test functions. In future research, we will combine our proposal with cooperative co-evolution to solve large-scale optimization problems. Finally, the estimation of a convergence point combined with the Gaussian sampling operator has a broad prospect to solve optimization problems and can be easily extended with other population-based evolutionary algorithms.

VI. ACKNOWLEDGEMENT

This work was supported by JSPS KAKENHI Grant Number JP20K11967.

REFERENCES

[1] M. A. Ahandani, N. P. Shirjoposh, and R. Banimahd, “Three modified versions of differential evolution algorithm for continuous optimization,” *Soft Computing*, vol. 15, no. 4, pp. 803–830, 2010.

[2] Z. Yang, K. Tang, and X. Yao, “Differential evolution for high-dimensional function optimization,” 2007, pp. 3523–3530.

[3] F. Wang, W. Wang, T. Feng, and H. Li, “Parallel test sheets generation using differential evolution algorithm with constraint effective encoding and either-or mutation,” in *2015 2nd International Conference on Information Science and Control Engineering*. IEEE, 2015, pp. 376–379.

[4] K. Deb, “Multi-objective optimisation using evolutionary algorithms: an introduction,” in *Multi-objective evolutionary optimisation for product design and manufacturing*. Springer, 2011, pp. 3–34.

[5] D. Wolpert and W. Macready, “No free lunch theorems for optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[6] M. Norman, P. Moscato, and L. Plata, “A competitive-cooperative approach to complex combinatorial search,” 1991.

[7] A. Abdolmaleki, B. Price, N. Lau, L. P. Reis, and G. Neumann, "Deriving and improving cma-es with information geometric trust regions," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2017, pp. 657–664.

[8] F. Fröhlich and P. K. Sorger, "Fides: Reliable trust-region optimization for parameter estimation of ordinary differential equation models," *PLOS Computational Biology*, vol. 18, no. 7, p. e1010322, 2022.

[9] N. Murata, R. Nishii, H. Takagi, and Y. Pei, "Analytical estimation of the convergence point of populations," in *2015 IEEE Congress on Evolutionary Computation, CEC 2015 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., Sep. 2015, pp. 2619–2624.

[10] J. Yu, Y. Li, Y. Pei, and H. Takagi, "Accelerating evolutionary computation using a convergence point estimated by weighted moving vectors," *Complex and Intelligent Systems*, vol. 6, 05 2019.

[11] J. Liang, B. Qu, P. Suganthan, and A. Hernández-Díaz, "Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization," *Technical Report 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China*, 01 2013.

[12] L. A. Rastrigin, "Convergence of random search method in extremal control of multi-parameter systems," *Avtomat.i Telemekh*, p. 1467–1473, 1963.

[13] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[14] R. Storn, "On the usage of differential evolution for function optimization," in *Proceedings of north american fuzzy information processing*. Ieee, 1996, pp. 519–523.

[15] L. B. Slobodkin, "The strategy of evolution," *American scientist*, vol. 52, no. 3, pp. 342–357, 1964.

[16] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948 vol.4.

TABLE III

CEC2013 SUITE: UNI=UNIMODAL, MULTI=MULTIMODAL, COMP=COMPOSITION

Func.	2-D					10-D					30-D				
	P1	P2	P1 with DE	P2 with DE	P1 with P2	P1	P2	P1 with DE	P2 with DE	P1 with P2	P1	P2	P1 with DE	P2 with DE	P1 with P2
f_1	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $>$ P2	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\approx$ P1	PSO $\approx$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\gg$ DE	P2 $\gg$ DE	P2 $\gg$ P1	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\approx$ P1	PSO $\approx$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_3	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\approx$ P1	PSO $\approx$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_4	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $>$ DE	P2 $>$ DE	P1 $\approx$ P2	PSO $\approx$ P1	PSO $\approx$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_5	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $>$ DE	P2 $>$ DE	P1 $\approx$ P2	PSO $\approx$ P1	PSO $\approx$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_6	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $>$ DE	P2 $>$ DE	P1 $\approx$ P2	PSO $\approx$ P1	PSO $\approx$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_7	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\approx$ P1	PSO $\approx$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_8	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2
f_9	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $>$ DE	P2 $>$ DE	P1 $\approx$ P2	PSO $\approx$ P1	PSO $\approx$ P2	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2
f_{10}	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\approx$ P1	PSO $\approx$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{11}	P1 $\approx$ PSO	P2 $>$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\approx$ GA	P2 $\approx$ GA	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{12}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{13}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{14}	P1 $\approx$ GA	P2 $\approx$ GA	P1 $\approx$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2	GA $\gg$ P1	GA $\gg$ P2	P1 $\approx$ DE	P2 $>$ DE	P1 $\approx$ P2
f_{15}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $>$ DE	P1 $\approx$ P2	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{16}	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2	P1 $\approx$ GA	P2 $\approx$ GA	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2	P1 $\approx$ ES	P2 $\approx$ ES	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2
f_{17}	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2
f_{18}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $>$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{19}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2	P1 $\gg$ PSO	P2 $>$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{20}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $>$ DE	P2 $>$ DE	P1 $\approx$ P2
f_{21}	P1 $\gg$ PSO	P2 $\gg$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $>$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{22}	P1 $\approx$ GA	P2 $\approx$ GA	P1 $\gg$ DE	P2 $\gg$ DE	P1 $>$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	GA $\gg$ P1	GA $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{23}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2
f_{24}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\approx$ ES	P2 $\approx$ ES	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{25}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $>$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\approx$ DE	P1 $\approx$ P2
f_{26}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	P1 $\approx$ GA	P2 $>$ GA	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\approx$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{27}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\gg$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\approx$ DE	P2 $\gg$ DE	P1 $\approx$ P2
f_{28}	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $\approx$ DE	P2 $\approx$ DE	P1 $\approx$ P2	P1 $\approx$ PSO	P2 $\approx$ PSO	P1 $>$ DE	P2 $\gg$ DE	P1 $\approx$ P2	PSO $\gg$ P1	PSO $\gg$ P2	P1 $\approx$ DE	P2 $\gg$ DE	P1 $\approx$ P2