Effects of Centralized Population Initialization in Differential Evolution

Hojjat Salehinejad and Shahryar Rahnamayan, *Senior Member, IEEE*
Department of Electrical, Computer, and Software Engineering University of Ontario Institute of Technology
Oshawa, Ontario, Canada
{hojjat.salehinejad, shahryar.rahnamayan}@uoit.ca

***Abstract*—Differential evolution (DE) is one of the highest performance, easy to implement, and low complexity population-based optimization algorithms. Population initialization plays an important role in finding better candidate solution and faster convergence of the population to a global optimum. It has been shown in the literature that large population sizes for large-scale problems necessarily does not show a statistically significant performance improvement over medium size population. In this paper, we emphasise on importance of population initialization and discuss effects of using centroid-based population initialization in DE, with focus on micro-DE (i.e. DE with small population size). Experimental results for high and low dimensional problems with small and standard population sizes on CEC Black-Box Optimization Benchmark problems 2015 (CEC-BBOB 2015) show centroid initialization can increase performance of DE algorithm, compared to the conventional initialization method.**

I. Introduction

Population-based algorithms are among popular global optimization methods [1]. Utilization of more than one individual in these methods helps the algorithm to have better exploration and collaboration capability to find optimal solutions. Differential evolution (DE) algorithm is one of the pioneer methods in this category. DE is easy to implement and modify [2]. It has presented extraordinary performance in solving challenging benchmark and real world problems [2], [3].

DE is consisted of four major steps which are population initialization, mutation, crossover, and selection. Contribution of these steps to overall algorithm performance is dependent on appropriate setting of control parameters. Population size, mutation scale factor, crossover rate, and mutation scheme are among the important ones to single out.

Enhanced population initialization is a key factor in performance of DE. A better initialization technique can enhance its exploration ability. On the other side, inappropriate initialization can degrade searching performance.

A parameter study of initialization methods for large-scale problems is provided in [4]. Many research works have been conducted on population initialization enhancement of DE. Most of the methods are focused on uniform initialization for low dimensional problems. Importance of initialization is studied in [5] for large-scale optimization. This work provides a parameter analyses and comparative study on different DE algorithms for functions with 905 and 1000 decision variables [5]. The obtained results show that utilizing large population sizes for large-scale problems necessarily does not provide statistically significant performance improvement.

The conventional way to deal with large-scale problems is increasing the population size as the problem dimension increases. However, the population size directly affects the computational cost of DE and developer should be aware of limits [5]. Small population size decreases the chance of exploration of promising regions on the landscape. A vectorized random mutation scale factor is proposed in [2] and [6] to enhance the exploration capability of DE algorithm with small population size. DE algorithm with a small population size is called micro-DE, which has much less computational cost comparing to standard population size DE. It has been utilized for 3D localization of sensor in [7].

In addition to the mutation scale factor [2], studies show that proper population initialization increases the probability of finding global solution [8], increases robustness [8], reduces the computational cost [9], and enhances the solution quality [10], [6]. In this paper, we visualize and discuss centroid-based population initialization for DE (CIDE) and its micro version. The idea of centralization is to initialize the population uniform randomly within a certain boundary of the original population boundary. For example, if the initial distribution of population is allowed within the range of $[a, b]$, the centroid method initializes them within a central percentile of the boundaries a and b. This technique starts the search from positions which are likely to be closer to the global optimal solution. The general assumption is that the closer is an individual to solution, its fitness value is better; this assumption is true for all non-deceptive problems.

Next section provides a review on initialization techniques utilized in DE. Section III provides a short introduction to DE. The centroid-based population initialization is studied in Section IV. The experimental results are discussed in Section V and finally the paper is concluded and future remarks are presented in Section VI.

II. Related Works

The initialization techniques in evolutionary algorithms are categorized in terms of randomness, compositionality, and generality [5].

Pseudo-random number generators (PRNGs) are the most widely used population initialization techniques [9], [10]. It is claimed that these methods are not the best available options for population initialization of DE when dealing with large-scale problems [4], [11]. Chaotic number generators

978-1-5090-4240-1/16/$31.00 ©2016 IEEE

(CNGs) improve the randomness and uniformity of the initial population [12]. Ergodicity, randomness, and unpredictability are the main characteristics of chaotic systems [13]. In order to produce a chaotic sequence, a proper map for the variable $x_{i,j}^k$ at iteration k is required, such as a tent map:

$$x_{i,d}^{k+1} = \begin{cases} \mu x_{i,d}^k & x_{i,d}^k < \frac{1}{2} \\ \mu(1 - x_{i,d}^k) & x_{i,d}^k \geq \frac{1}{2} \end{cases}, \quad (1)$$

where i and d represent the individual and dimension, respectively. If the parameter $\mu = 2$ and $x_{i,d}^0 \in (0, 1)$, this map produces chaotic sequences [4].

The DE is hybridized with non-linear simplex method with uniform random numbers for population initialization, named NSDE [14]. This method generates N_P individuals and uses Nelder-Mead Simplex (NMS) to generate the same number of individuals from previously generated individuals [14]. In [15], the NM-DE method generates N_P random individuals $\mathbf{X}_i$, [16], and then computes centroid of the top Q individuals as:

$$\bar{\mathbf{X}} = \sum_{q=1}^{Q} \frac{\mathbf{X}_q}{Q}. \quad (2)$$

Then, $N_P - Q$ new points are generated using the simplex method and the centroid $\bar{\mathbf{X}}$ [15], [16]. This model applies a one point shrinking until the point gets a better value. The same idea is utilized in [16] for the JADE algorithm, an adaptive DE algorithm, by generating $3 \times N_P$ random individuals and calculating their centroids as:

$$\mathbf{X}_i = \frac{\mathbf{X}_1 + \mathbf{X}_2 + \mathbf{X}_3}{3}, \quad (3)$$

until N_P centroids are computed to use as initial population [16].

Deterministic uniform number generator methods focus on uniform distribution rather than randomness of points on the problem search space. Some approaches come with theoretical upper-bounds on the non-uniformity of the points [17]. The population is sparse in high-dimensional search spaces. It was assumed that uniform distribution of initial points can help the algorithm [10], [18]; however, the practice shows that uniform distribution of initial points may not perform as expected in high dimensional search spaces [19], [8], [20].

Some initialization methods evaluate an objective function to initialize the population [3]. Some of these methods are quadratic interpolation [21], non-linear local search [22], [23], and smart sampling [24]. A local-global selection method to generate high quality initial individuals for job-shop scheduling is proposed in [25]. Another population initialization method using local search method based on hill-climbing technique is proposed in [26]. A comprehensive review on these methods is presented in [5].

Prior knowledge with a low computational cost can assist to generate proper initial points. Opposition-based learning (OBL) has introduced a new direction in machine learning and particularly population-based algorithms [3], [27]. The simple but effective OBL method in population initialization considers a set of randomly generated points and their opposites. It evaluates the generated points using the corresponding objective function and selects a subset of fitted points as the initial population [28]. This model is initially developed for DE algorithm, called opposition-based DE (ODE). In [29], [28], Monte-Carlo and mathematical methods are used to discuss why opposite points are working better than the uniform random points.

The OBL computes opposite of an individual i with respect to the original individual value $x_{i,d}$ as:

$$\tilde{x}_{i,d} = x_d^{min} + x_d^{max} - x_{i,d} \quad (4)$$

where x_d^{min} and x_d^{max} are the lower and upper boundaries on dimension d, respectively [3]. Quasi-opposite point is another version of OBL scheme for population initialization and generation jumping, which randomly generates points between a middle point and its corresponding opposite [30].

The idea of using centroid in DE and simulated annealing is proposed in the literature [31], [19]. The idea of utilizing centroid to calculate opposite points is utilized in ODE recently, called centroid opposition-based computation (COBC) [32], [33]. The COBC uses the center of gravity of the population, instead of min and max points, to consider the entire population in its opposition scheme. Definition of centroid is "the point where the centre of mass lies in a uniform body" [33]. By considering the entire population of DE as a discrete body, the unit mass is distributed. Experiments show that centroid of population has special characteristics which strengthen the learning process of the algorithm and also involve the whole population in a better way [32].

III. Differential Evolution

DE is an effective population-based optimization algorithm to solve a global optimization problem formalized as:

$$Minimize\ f(\mathbf{X}), \mathbf{X} = [x_1, ..., x_D] \in \boldsymbol{R}, \quad (5)$$

subject to:

$$g_j(\mathbf{X}) \leq 0,\ for\ j = 1, ..., P, \quad (6)$$

where $x_d^{min} \leq x_d \leq x_d^{max}$ for dimension $d = 1, ..., D$, P is the number of constraints, and x_d^{min} and x_d^{max} indicate the lower and upper bounds of the variable x_d, respectively. DE operates through four stages, described below, in order to find a desired optimal solution.

A. Population Initialization

The population initialization phase initiates the search towards the global optimum by having NP stage number of individuals in the population, D dimensional, uniform randomly generated candidate solutions, which are known as initial vectors. Population is subject to change over a limited number of generations. It is customary to denote the population i at generation g such as:

$$\mathbf{X}_{i,g} = [\mathbf{x}_{1,i,g}, ..., \mathbf{x}_{D,i,g}], \quad (7)$$

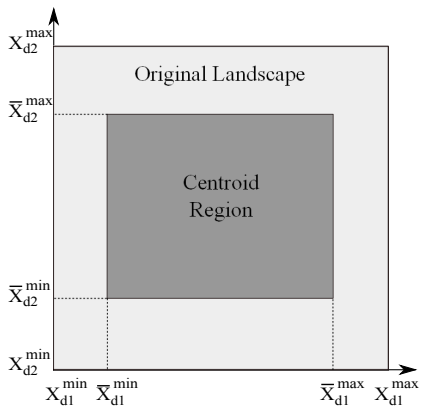


Fig. 1: Centroid boundaries on a two dimensional search space. Dimensions are denoted by $d1$ and $d2$. The original search space (light grey square) refers to the original boundaries of the dimensions. The centroid region (dark grey square) refers to the centroid boundaries of the dimensions.

where $g = 0, ..., G$. During the initialization stage, the vector $x_{d,i,0}$ is initialized according to the following equation:

$$x_{d,i,0} = x_d^{min} + rand_{i,d}(0,1) \times [x_d^{max} - x_d^{min}], \quad (8)$$

where $d = 1, .., D$, $i = 1, ..., N_P$, $rand(0,1)$ generates a uniform random number in $[0,1]$, and x_d^{min} and x_d^{max} are the lower and upper boundaries of the variable x_d, respectively.

B. Mutation Operator

A mutation operator generates a vector known as a donor vector $\mathbf{V}_{i,g}$ for each vector in the current population, identified as a target vector. Although there are variant DE-mutation schemes [3], [2], the classical version is DE/rand/1 given as

$$\mathbf{V}_{i,g} = \mathbf{X}_{r_1,g}^i + F(\mathbf{X}_{r_2,g}^i - \mathbf{X}_{r_3,g}^i), \quad (9)$$

where $i = 1, ..., N_P$. Here, the indices r_1, r_2, r_3 are mutually exclusive integers chosen randomly from the set $\{1, ..., N_P\}$. Furthermore, the value of the amplification factor F of the difference vector typically lies in the interval $(0, 2]$.

C. Crossover Operator

By shuffling a donor vector with its associated target vector to enhance the potential diversity of the population, this phase results in a vector known as a trial vector $\mathbf{U}_{i,g}$ defined as follows:

$$U_{d,i,g} = \begin{cases} v_{d,i,g}, & rand_{i,d}(0,1) \leq Cr \text{ or } d = rand_d \\ x_{d,i,g}, & otherwise \end{cases}, \quad (10)$$

where $rand_{i,d}(0,1)$ is the d^{th} uniformly distributed random number generated for the i^{th} trial vector, $Cr \in (0,1)$ is a constant crossover rate, and $rand_d \in \{1, ..., D\}$ is a random integer number, where ensures $\mathbf{U}_{i,g}$ inherits at least one component from $\mathbf{V}_{i,g}$.

D. Selection

Finally, this step leads to a new generation $g + 1$, which is derived by having made the selection either to retain the old solution $x_{i,g}$ or introduce a new candidate solution $\mathbf{U}_{i,g}$ instead. For a minimization problem, the mentioned greedy selection is defined as follows:

$$\mathbf{X}_{i,g+1} = \begin{cases} \mathbf{U}_{i,g}, & f(\mathbf{U}_{i,g}) \leq f(\mathbf{X}_{i,g}) \\ \mathbf{X}_{i,g}, & f(\mathbf{U}_{i,g}) > f(\mathbf{X}_{i,g}) \end{cases}, \quad (11)$$

where $i = 1, ..., N_P$. A comprehensive survey about DE is presented in [34].

Algorithm 1: Centroid-based Initialized Differential Evolution (CIDE)

Procedure CIDE
$g = 0$ // generation counter
// Initial Population Generation
for $i = 1 \rightarrow N_P$ **do**
  for $d = 1 \rightarrow D$ **do**
    $\bar{x}_d^{min} = x_d^{min} + \frac{1-C}{2}(x_d^{max} - x_d^{min})$
    $\bar{x}_d^{max} = x_d^{max} - \frac{1-C}{2}(x_d^{max} - x_d^{min})$
    $\mathbf{X}_{i,d} = \bar{x}_d^{min} + rand(0,1) \times (\bar{x}_d^{max} - \bar{x}_d^{min})$
  end for
  $\mathbf{P}_i^g = \mathbf{X}_i$
end for
// End of Initial Population Generation
while $(|BFV - VTR| > EVTR \ \& \ NFC < NFC_{Max})$ **do**
  for $i = 1 \rightarrow N_P$ **do**
  // Mutation
    Select three random population vectors from $\mathbf{P}^g$ *where* $(i_1 \neq i_2 \neq i_3 \neq i)$
    $F = 0.5$ //mutation scale factor
    for $d = 1 \rightarrow D$ **do**
      $\mathbf{V}_{i,d} = \mathbf{X}_{i_1,d} + F(\mathbf{X}_{i_2,d} - \mathbf{X}_{i_3,d})$
    end for
  // End of Mutation
  // Crossover
    for $d = 1 \rightarrow D$ **do**
      if $rand(0,1) < Cr$ or $d_{rand} = d$ **then**
        $U_{i,d} = V_{i,d}$
      else
        $U_{i,d} = x_{i,d}$
      end if
    end for
  // End of Crossover
  // Selection
    if $f(\mathbf{U}_i) \leq f(\mathbf{X}_i)$ **then**
      $\mathbf{X}_i' = \mathbf{U}_i$
    else
      $\mathbf{X}_i' = \mathbf{X}_i$
    end if
  // End of Selection
  end for
  $\mathbf{X}_i = \mathbf{X}_i', \forall i \in \{1, ..., N_P\}$
  $g = g + 1$
  $\mathbf{P}^g = \{\mathbf{X}_1, ..., \mathbf{X}_{N_P}\}$
end while

IV. Centralized Population Initialization

The most common initialization method in DE algorithm is random initialization with uniform distribution between the lower x_d^{min} and upper x_d^{max} boundaries of each dimension d. Population initialization close to the center of interval can accelerate convergence of algorithm to a solution, particularly in large-scale problems. The centroid interval C (as a portion of whole interval) is selected from the range $(0, 1)$. Figure 1

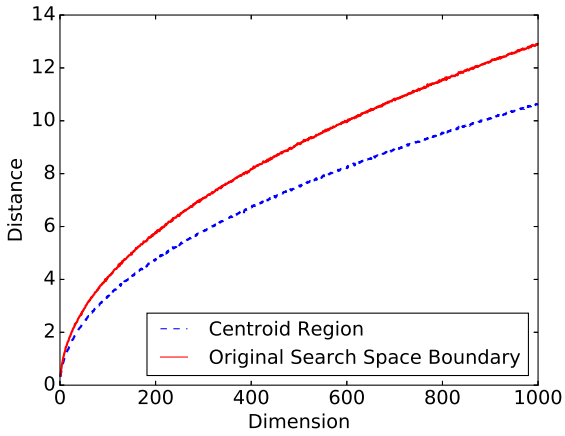


Fig. 2: Monte-Carlo experiment, comparing the average distance between points generated uniformly in centroid region and original region from the solution. The boundary of the original landscape is [0,1].

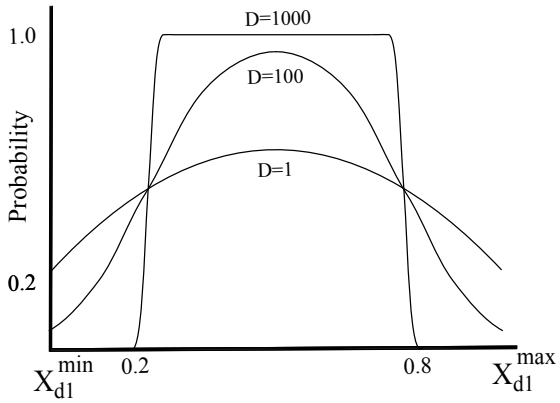


Fig. 3: The graphs of Monte-Carlo simulations which present the probability of closeness of candidate-solution (by considering set of points corner-to-corner with the step-size of 0.01 in the search space) to an unknown solution in the interval $[X_{d1}^{min}, X_{d1}^{max}]$, for various dimensions [35].

shows the lower $\bar{x}_d^{min}$ and upper $\bar{x}_d^{max}$ boundaries of the centroid interval, which are calculated as:

$$\bar{x}_d^{min} = x_d^{min} + \frac{1-C}{2}(x_d^{max} - x_d^{min}) \tag{12}$$

and

$$\bar{x}_d^{max} = x_d^{max} - \frac{1-C}{2}(x_d^{max} - x_d^{min}). \tag{13}$$

Pseudo-code of the centroid population initialization is presented in Algorithm 1. It is easy to implement, because just modifying the conventional DE algorithm at the initialization step is required. A Monte-Carlo simulation is conducted for a search space range of [0,1] and a variety of search space dimensions $d \in \{1, 2, ..., 1000\}$. The global solution is an unknown random point in the search space (similar to a black-box problem). The distance in original search space refers to the average distance between distributed individuals in the whole search space from the global solution. Similarly, for the centroid region it refers to the average distance between the uniform individuals generated in the central region and the global solution. The Monte-Carlo simulation in Figure 2 shows that as the dimensionality of the search space increases, the average distance of random points generated in the centroid region is less than the distance of the uniform points generated in the original search space. Generating of initial population within the centroid region enhances the search process of the optimizer.

It is reported in [35], [31], and [19] that probability of closeness to an unknown solution increases by moving the individuals closer to the center of the search space. Monte-Carlo simulation results presented in Figure 3 for dimensions $d \in \{1, 100, 1000\}$ show that as the search space dimensionality increases, the probability of closeness to the solution improves as well [31]. This improvement is in its highest value in the range [0.2,0.8] in a [0,1] search space interval. For high dimensions, the probability is approaching to one and the probability curve gets a flat shape [31], [19].

We can use Monte-Carlo method to measure the ratio of generated uniform random points on a search space and the centroid region. For 10^6 uniform randomly generated points over the whole search space, the results show that $40, 19, ..., 1, 0, ..., 0$ percentile of the points for problem dimensionalities $d = 2, 3, ..., 15, 16, ..., 1000$, are generated in the centroid region. The percentile drops to zero at $d = 16$. This shows that as the dimensionality of search space increases, the chance of having points in the centroid region is approaching zero exponentially. This is due to the volume of centroid region hyper-cube, which is much smaller than the original search space. By considering V_c as the volume of centroid region hyper-cube and V_o as the volume of original search space hyper-cube, we have:

$$\frac{V_c}{V_o} = \frac{\prod_{d=1}^{D}(\bar{x}_d^{max} - \bar{x}_d^{min})}{\prod_{d=1}^{D}(x_d^{max} - x_d^{min})}, \tag{14}$$

where as the search space dimension increases $d \to \infty$ (i.e. toward large-scale search space), the ratio $V_c/V_o \to 0$. For example in a $D = 1000$ dimensional search space, if $[x_d^{min}, x_d^{max}] = [0, 1]$ and $[\bar{x}_d^{min}, \bar{x}_d^{max}] = [0.2, 0.8]$, then $V_c/V_o = (1.41e-222)/1 = 1.41e-222 \approx 0$. The proposed centroid-based initialization approach forces initialization population to have individuals in the centroid-region, which are much closer to optimal solution compared to other uniform individuals generated in the whole search space, particularly in large-scale search spaces.

The centroid based initialization forces the solution toward the centre of the domain. When the problem dimensionality increases obviously exploitative approaches in DE perform better than exploratory ones. This is because there is not enough budget for achieving any seriously promising search direction. Hence, one of the best strategies is to start exploiting from the beginning. The budget exhaustion will occur before

premature convergence. In other words, the search space increases exponentially with the dimensionality while the budget we assign grows linearly with it.

V. Experimental Results

In this section we evaluate the center-based population initialization using the black-box optimization benchmark (CEC-BBOB-2015) functions given at CEC-2015 [36]. The benchmarking consists of 24 noise-free functions in five classes which are separable functions ($f_1 - f_5$), functions with low or moderate conditioning ($f_6 - f_9$), functions with high conditioning and uni-modal ($f_{10} - f_{14}$), multi-modal functions with adequate global structure ($f_{15} - f_{19}$), and multi-modal functions with weak global structure ($f_{20} - f_{24}$). The functions are shifted and the global solution is not at the zero point (middle of interval).

The experiments are conducted for dimensions $D = \{10, 100, 1000\}$, crossover rate $Cr = 0.9$, maximum number of function calls $NFC_{max} = 10,000D$, and error-value-to-reach $EVTR = 10e - 9$. The population size is set to $N_P = \{6, 100\}$. The portion of centroid interval is set to $C = \{0.4, 0.6, 1\}$.

The experiments are conducted for 30 independent runs per function. The average of error value and corresponding standard deviation are reported in Tables I to III. Summary of the tables is presented in Table IV. The Wilcoxon statistical test indicates the significant result, where '+' symbol demonstrates significance of DE with default initialization interval (i.e. DE_{100}) over proposed method with initialization intervals [-40,40] and [-60,60] (i.e. CIDE_{40} and CIDE_{60}, respectively).

The results in Table IV show that for small population size $N_P = 6$ and low problem dimension $D = 10$ the CIDE_{60} and CIDE_{40} have better performance than DE_{100} with 21 success out of 24 functions. We can observe similar performance for $D = \{100, 1000\}$. As the population increases to 100 for $D = 10$, we observe 14 neutral and 10 success counts. This show as the population size increases for low dimensional problems, the initialization interval has less effect on the performance. This is mostly due to the increased diversity in the population by large population size. This is while as the dimensionality increases, we observe a high number of success for both small and large population sizes. Because in large dimentions the population size has minor impact on the performance and exploitation has better result that exploration [2].

Performance plots of the DE with initialization interval of [-100,100] and CIDE with initialization intervals of [-40,40] and [-60,60] in terms of average of error versus number of function calls are presented in Figure 4. The first function from each function class is selected for visualization, i.e. f_1, f_6, f_{10}, f_{15}, and f_{20}. The population size is $N_P = \{6, 100\}$ and the problem dimension is $d \in \{10, 100, 1000\}$. The plots are average of 30 independent runs. For some functions such as f_{15} we observe almost flat error. This is mostly because of the function structure multi-modal functions with adequate global structure and pre-mature convergence of population. We can observe performance of the CIDE_{40} and CIDE_{60} for other functions. For example, f_{20} for $D = 1000$ show fast convergence of the CIDE_{40} to a solution while it takes the whole number of function calls for the DE_{100} to get close to the found solution by centralized initialization.

VI. Conclusions and Future Work

Population initialization, mutation, crossover, and selection are major steps of differential evolution (DE) algorithms. Proper setting of parameters in each stage has a direct impact on performance of the algorithm. Conventionally, large population sizes are utilized for large-scale problem. However, results show that this not only increases the computational cost but also has a negative effect on the performance of the algorithm. Our experiments show that a small population size has competitive performance comparing to a standard population size for very large-scale problems. Initializing population within a specific interval of the dimension boundaries enhances performance of algorithm at the initial evolution stages. This centroid-based initialization technique has achieved superior results versus conventional DE algorithm.

For the small population size six, as the search space dimension increases, the centroid-based initialization DE (CIDE) method has better performance than the conventional uniform random initialization approach. As the problem dimensionality increases, regardless of the population size, CIDE shows better performance. Regarding large population sizes, the CIDE method has superior performance by almost being successful in all the benchmark problems. The DE algorithm with large population size has competitive performance with CIDE with small population size for low-dimensional problems. However, the CIDE has superior performance in large-scale problems with the same setting.

For future works, it is important to study different centroid intervals and effect of allowing more number of function calls for evolution. The centroid initialization method should be compared with other state-of-the-art methods for performance assessment. The initialization should be tested in a different class of test problems in order to analyze its limitations and challenges. This technique is applicable for different population-based algorithm, with the least required modification in the algorithm. A comparison against similar methods (other than uniform initialization) is necessary in the future.

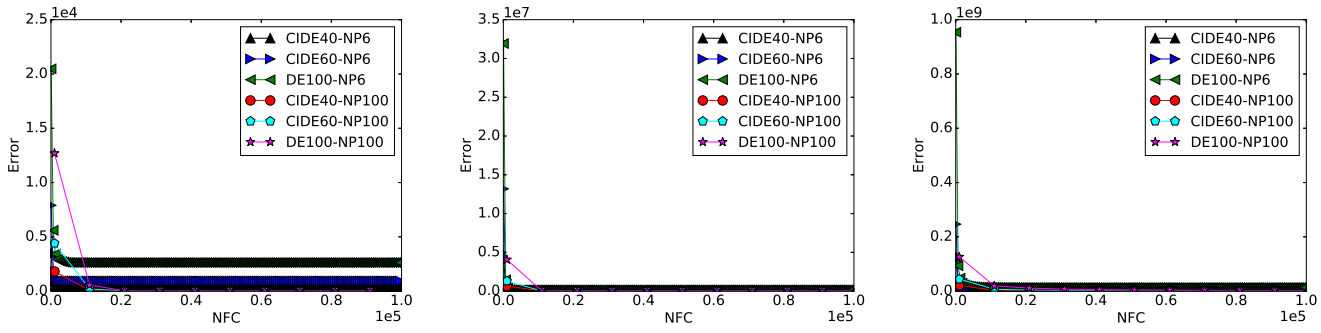
References

[1] H. Salehinejad and S. Talebi, "Dynamic fuzzy logic-ant colony system-based route selection system," *Applied Computational Intelligence and Soft Computing*, vol. 2010, 2010.

[2] H. Salehinejad, S. Rahnamayan, H. R. Tizhoosh, and S. Y. Chen, "Micro-differential evolution with vectorized random mutation factor," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 2055–2062.

[3] S. Rahnamayan, H. R. Tizhoosh, and M. Salama, "Opposition-based differential evolution," *Evolutionary Computation, IEEE Transactions on*, vol. 12, no. 1, pp. 64–79, 2008.

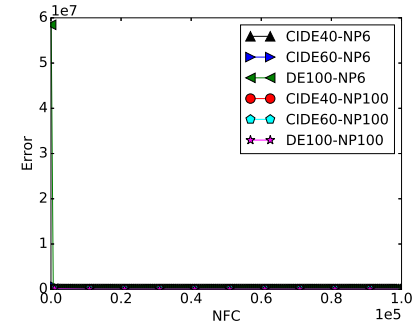
[4] B. Kazimipour, X. Li, and A. Qin, "Effects of population initialization on differential evolution for large scale optimization," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 2404–2411.



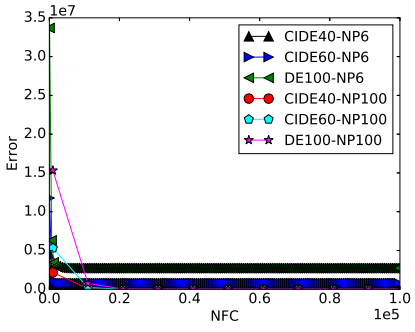
(a) $f_1; D = 10$

(b) $f_6; D = 10$

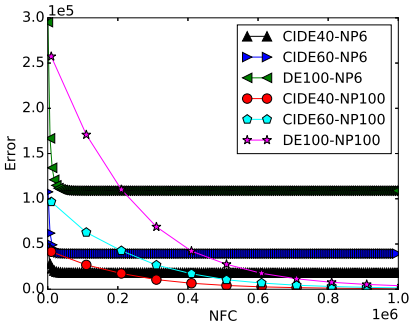
(c) $f_{10}; D = 10$



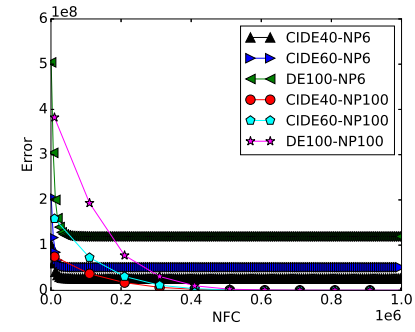
(d) $f_{15}; D = 10$



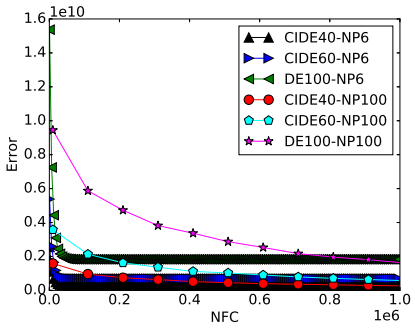
(e) $f_{20}; D = 10$



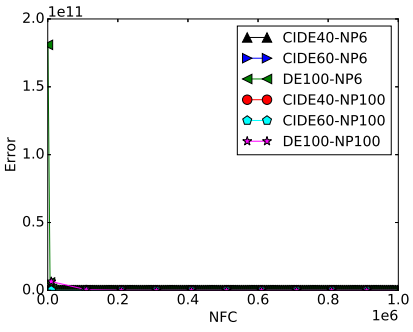
(f) $f_1; D = 100$



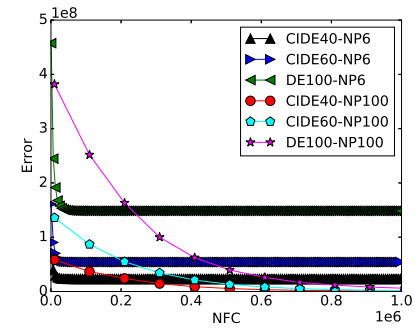
(g) $f_6; D = 100$



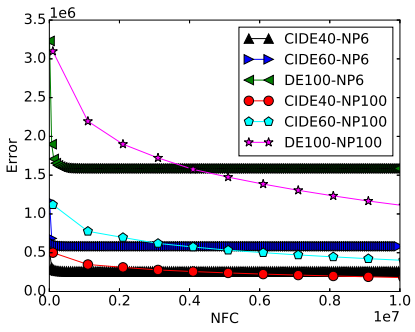
(h) $f_{10}; D = 100$



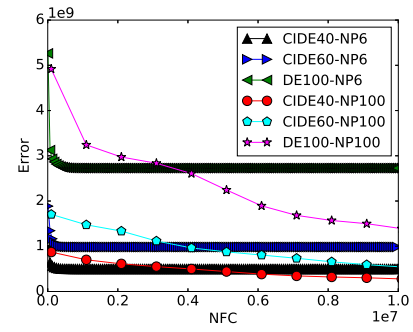
(i) $f_{15}; D = 100$



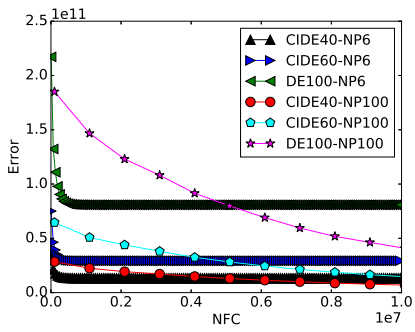
(j) $f_{20}; D = 100$



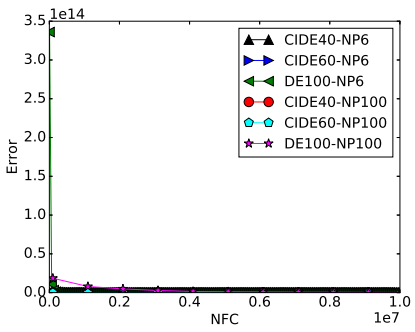
(k) $f_1; D = 1000$



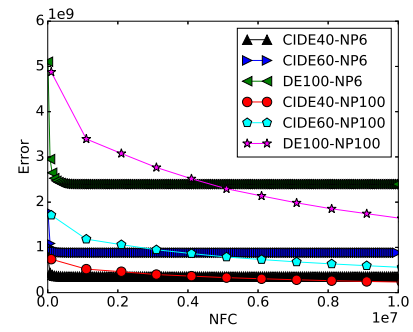
(l) $f_6; D = 1000$



(m) $f_{10}; D = 1000$



(n) $f_{15}; D = 1000$



(o) $f_{20}; D = 1000$

Fig. 4: Error versus number of function calls for the CIDE with centroid intervals [-40,40] and [-60,60] and DE with default interval [-100,100] for objective functions f_1,f_6,f_{10},f_{15}, and f_{20}. Population size is $\mathbf{N_P}$=**{6,100}**, dimensions are **D=10,100,1000**.

TABLE I: Error and error standard deviation for population initialization intervals [-40,40], [-60,60], and [-100,100]; **D=10** and $\mathbf{N_P}$=**{6,100}**.

F	N_P=6					N_P=100				
	[-100,100]	[-60,60]		[-40,40]		[-100,100]	[-60,60]		[-40,40]	
	Error	Error	W	Error	W	Error	Error	W	Error	W
1	2.63e+03±1.95e+03	9.35e+02±8.29e+02	−	3.94e+02±3.15e+02	−	6.00e-09±2.93e-09	5.51e-09±2.59e-09	=	5.55e-09±2.41e-09	=
2	1.37e+07±2.71e+07	9.96e+06±2.61e+07	=	7.42e+06±1.67e+07	=	6.59e-06±4.53e-06	2.29e-06±2.44e-06	−	9.72e-07±7.07e-07	−
3	4.92e+04±5.92e+04	9.12e+03±1.39e+04	−	4.06e+03±4.07e+03	−	9.79e+00±4.51e+00	9.27e+00±3.70e+00	=	8.75e+00±3.95e+00	=
4	2.70e+05±2.28e+05	7.98e+04±7.74e+04	−	3.07e+04±3.23e+04	−	1.08e+01±4.41e+00	9.54e+00±5.23e+00	=	8.60e+00±3.76e+00	−
5	1.15e+01±2.40e+01	1.70e+01±3.20e+01	=	1.63e+01±3.43e+01	=	1.00e-08±0.00e+00	1.00e-08±0.00e+00	=	1.00e-08±0.00e+00	=
6	1.61e+05±4.82e+05	5.45e+03±3.17e+03	=	3.78e+03±2.52e+03	=	1.43e+02±1.38e+02	4.71e+01±2.54e+01	−	2.56e+01±1.34e+01	−
7	1.39e+04±1.05e+04	3.44e+03±2.54e+03	−	1.52e+03±1.18e+03	−	1.03e+00±5.89e-01	5.85e-01±3.36e-01	−	7.95e-01±4.77e-01	=
8	1.63e+08±2.34e+08	1.69e+07±1.48e+07	−	7.74e+06±1.22e+07	−	5.44e+01±1.19e+02	2.69e+01±5.13e+01	=	8.56e+00±1.76e+01	−
9	8.73e+07±1.06e+08	9.25e+06±1.82e+07	−	1.73e+06±2.58e+06	−	5.89e+03±1.75e+04	7.03e+02±1.96e+03	=	2.79e+02±1.03e+03	=
10	1.31e+07±1.34e+07	3.86e+06±4.08e+06	−	1.32e+06±1.79e+06	−	1.98e+06±1.36e+06	6.21e+05±4.34e+05	−	2.43e+05±1.41e+05	−
11	1.93e+04±1.23e+04	7.43e+03±3.76e+03	−	2.90e+03±2.19e+03	−	1.46e+04±6.52e+03	4.84e+03±1.87e+03	−	2.18e+03±9.81e+02	−
12	1.03e+10±1.08e+10	2.71e+09±2.51e+09	−	1.35e+09±1.17e+09	−	6.34e+05±1.23e+06	5.65e+04±1.09e+05	−	9.00e+03±1.69e+04	−
13	8.12e+03±3.60e+03	4.77e+03±1.60e+03	−	3.23e+03±1.64e+03	−	1.01e+01±1.24e+01	7.80e+00±1.26e+01	=	6.23e+00±5.88e+00	=
14	1.34e+03±1.13e+03	3.38e+02±2.71e+02	−	1.64e+02±1.57e+02	−	3.63e+01±3.88e+01	2.61e+00±4.26e+00	−	4.40e-01±1.32e+00	−
15	3.19e+04±3.09e+04	9.20e+03±9.04e+03	−	4.05e+03±3.14e+03	−	4.18e+01±5.85e+00	3.93e+01±7.51e+00	=	4.08e+01±7.56e+00	=
16	1.39e+03±1.30e+03	3.84e+02±2.82e+02	−	1.48e+02±2.12e+02	−	1.41e+01±2.57e+00	1.42e+01±2.30e+00	=	1.32e+01±2.40e+00	=
17	3.70e+04±5.08e+04	8.16e+03±6.93e+03	−	1.84e+03±2.96e+03	−	1.43e-01±2.50e-01	7.52e-02±1.47e-01	=	1.01e-01±1.99e-01	=
18	3.68e+04±3.68e+04	9.48e+03±1.12e+04	−	2.57e+03±2.26e+03	−	1.46e+00±1.96e+00	1.13e+00±1.15e+00	=	8.12e-01±8.15e-01	=
19	1.34e+04±1.88e+04	4.53e+03±6.67e+03	−	4.56e+02±7.72e+02	−	4.10e+00±1.53e+00	3.68e+00±9.17e-01	=	3.42e+00±8.99e-01	−
20	2.71e+06±2.78e+06	7.79e+05±6.48e+05	−	2.77e+05±1.72e+05	−	7.48e-01±4.17e-01	6.98e-01±3.26e-01	=	5.85e-01±2.45e-01	=
21	1.61e+03±1.32e+03	4.58e+02±3.49e+02	−	2.29e+02±1.82e+02	−	4.49e+00±1.93e+00	5.21e+00±6.14e+00	=	1.54e+00±2.28e-01	−
22	9.99e+02±9.83e+02	3.66e+02±3.99e+02	−	2.28e+02±2.11e+02	−	7.41e+00±1.03e+01	7.23e+00±1.06e+01	=	5.40e+00±7.40e+00	=
23	1.30e+03±1.23e+03	4.69e+02±3.89e+02	−	1.54e+02±1.74e+02	−	2.08e+00±3.76e-01	2.00e+00±3.13e-01	=	1.91e+00±3.75e-01	=
24	1.28e+07±1.26e+07	4.33e+06±4.50e+06	−	1.40e+06±1.41e+06	−	4.83e+01±7.35e+00	4.81e+01±5.96e+00	=	4.53e+01±7.46e+00	=

TABLE II: Error and error standard deviation for population initialization intervals [-40,40], [-60,60], and [-100,100]; **D=100** and $\mathbf{N_P}$=**{6,100}**.

F	N_P=6					N_P=100				
	[-100,100]	[-60,60]		[-40,40]		[-100,100]	[-60,60]		[-40,40]	
	Error	Error	W	Error	W	Error	Error	W	Error	W
1	1.09e+05±1.86e+04	3.94e+04±4.98e+03	−	1.83e+04±2.29e+03	−	3.81e+03±1.14e+03	1.45e+03±3.55e+02	−	6.12e+02±1.37e+02	−
2	3.58e+09±1.07e+09	1.29e+09±5.30e+08	−	6.35e+08±2.24e+08	−	2.20e+07±7.08e+06	8.62e+06±3.79e+06	−	3.61e+06±1.37e+06	−
3	2.37e+09±4.64e+09	1.70e+07±1.58e+07	−	1.49e+06±1.37e+06	−	2.87e+05±1.16e+05	4.63e+04±6.57e+03	−	1.56e+04±4.81e+03	−
4	1.58e+07±2.69e+06	4.81e+06±9.47e+05	−	1.83e+06±3.08e+05	−	3.47e+05±9.50e+04	1.03e+05±4.02e+04	−	2.53e+04±1.16e+04	−
5	3.32e+03±6.80e+02	2.26e+03±3.30e+02	−	1.72e+03±3.73e+02	−	1.00e-08±0.00e+00	1.00e-08±0.00e+00	=	1.00e-08±0.00e+00	=
6	1.19e+08±3.73e+07	5.09e+07±1.67e+07	−	2.64e+07±8.58e+06	−	3.64e+05±8.49e+04	1.57e+05±4.42e+04	−	6.96e+04±1.43e+04	−
7	7.40e+05±1.19e+05	2.54e+05±5.12e+04	−	1.08e+05±2.30e+04	−	9.68e+04±2.95e+04	3.92e+04±7.45e+03	−	1.72e+04±3.23e+03	−
8	7.88e+10±1.95e+10	1.02e+10±2.60e+09	−	2.17e+09±3.93e+08	−	1.12e+09±4.02e+08	1.19e+08±4.85e+07	−	2.74e+07±1.13e+07	−
9	7.66e+10±2.51e+10	9.91e+09±2.03e+09	−	1.86e+09±5.50e+08	−	3.23e+08±1.57e+08	4.72e+07±2.14e+07	−	1.04e+07±5.39e+06	−
10	1.83e+09±6.90e+08	6.90e+08±2.15e+08	−	2.89e+08±8.63e+07	−	1.62e+09±5.14e+08	5.76e+08±1.59e+08	−	2.66e+08±8.30e+07	−
11	3.92e+05±1.15e+05	1.13e+05±3.57e+04	−	5.49e+04±1.52e+04	−	3.41e+05±5.01e+04	1.09e+05±1.88e+04	−	5.10e+04±8.54e+03	−
12	6.20e+17±1.92e+18	6.32e+14±1.33e+15	=	6.81e+12±1.07e+13	=	2.21e+13±6.54e+13	6.98e+11±8.79e+11	=	8.32e+10±1.11e+11	=
13	6.42e+04±5.18e+03	3.91e+04±3.11e+03	−	2.56e+04±2.04e+03	−	1.76e+04±1.85e+03	1.05e+04±1.25e+03	−	6.64e+03±7.71e+02	−
14	6.52e+04±2.02e+04	1.69e+04±7.01e+03	−	6.82e+03±2.45e+03	−	3.63e+04±1.55e+04	6.89e+03±3.69e+03	−	2.63e+03±1.37e+03	−
15	5.00e+07±6.57e+07	2.18e+06±1.60e+06	−	3.93e+05±2.06e+05	−	1.15e+06±4.56e+05	2.08e+05±6.84e+04	−	4.81e+04±1.48e+04	−
16	8.99e+03±1.26e+03	2.77e+03±5.23e+02	−	1.03e+03±1.89e+02	−	2.58e+02±6.86e+01	1.27e+02±2.06e+01	−	9.80e+01±1.52e+01	−
17	2.78e+06±1.17e+06	5.14e+05±1.94e+05	−	1.45e+05±4.29e+04	−	4.61e+05±3.17e+05	2.41e+04±1.61e+04	−	1.37e+03±9.67e+02	−
18	3.07e+06±1.21e+06	7.00e+05±2.15e+05	−	1.77e+05±4.04e+04	−	9.41e+05±5.12e+05	6.07e+04±3.64e+04	−	3.49e+03±1.99e+03	−
19	1.99e+06±6.28e+05	2.26e+05±6.41e+04	−	4.88e+04±1.79e+04	−	8.28e+03±3.62e+03	1.06e+03±5.20e+02	−	2.06e+02±6.15e+01	−
20	1.49e+08±2.25e+07	5.43e+07±1.07e+07	−	2.31e+07±4.43e+06	−	5.92e+06±1.45e+06	1.65e+06±4.86e+05	−	5.76e+05±1.44e+05	−
21	8.49e+04±1.09e+04	2.73e+04±4.26e+03	−	9.61e+03±1.54e+03	−	1.56e+03±6.74e+02	3.53e+02±2.27e+02	−	1.32e+02±2.69e+01	−
22	9.06e+04±1.24e+04	2.56e+04±3.52e+03	−	9.18e+03±1.67e+03	−	1.55e+03±5.95e+02	3.46e+02±1.53e+02	−	1.31e+02±3.02e+01	−
23	8.32e+04±1.18e+04	2.62e+04±4.09e+03	−	9.13e+03±1.74e+03	−	1.45e+03±6.86e+02	2.80e+02±1.43e+02	−	5.39e+01±3.17e+01	−
24	8.44e+08±1.21e+08	2.56e+08±3.75e+07	−	9.72e+07±1.78e+07	−	1.55e+07±6.55e+06	3.06e+06±1.34e+06	−	4.60e+05±2.99e+05	−

[5] B. Kazimipour, X. Li, and A. Qin, "A review of population initialization techniques for evolutionary algorithms," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 2585–2592.

[6] H. Salehinejad, "Micro-differential evolution: Diversity enhancement and comparative study," Ph.D. dissertation, University of Ontario Institute of Technology, 2014.

[7] H. Salehinejad, R. Zadeh, R. Liscano, and S. Rahnamayan, "3d localization in large-scale wireless sensor networks: A micro-differential evolution approach," in *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*. IEEE, 2014, pp. 1824–1828.

[8] R. W. Morrison, "Dispersion-based population initialization," in *Genetic and Evolutionary ComputationGECCO 2003*. Springer, 2003, pp. 1210–1221.

[9] S. Kimura and K. Matsumura, "Genetic algorithms using low-discrepancy sequences," in *Proceedings of the 7th annual conference on Genetic and evolutionary computation*. ACM, 2005, pp. 1341–1346.

[10] Z. Ma and G. A. Vandenbosch, "Impact of random number generators on the performance of particle swarm optimization in antenna design," in *Antennas and Propagation (EUCAP), 2012 6th European Conference on*. IEEE, 2012, pp. 925–929.

[11] B. Kazimipour, X. Li, and A. Qin, "Initialization methods for large scale global optimization," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*. IEEE, 2013, pp. 2750–2757.

[12] M. Lanza, I. Barriuso, L. Valle, M. Domingo, J. Pérez, J. Basterrechea *et al.*, "Comparison of different pso initialization techniques for high dimensional search space problems: A test with fss and antenna arrays," in *Antennas and Propagation (EUCAP), Proceedings of the 5th European Conference on*. IEEE, 2011, pp. 965–969.

[13] Y. Gao and Y.-J. Wang, "A memetic differential evolutionary algorithm for high dimensional functions' optimization," in *Natural Computation, 2007. ICNC 2007. Third International Conference on*, vol. 4. IEEE, 2007, pp. 188–192.

[14] M. Ali, M. Pant, and A. Abraham, "Simplex differential evolution," *Acta*

TABLE III: Error and error standard deviation for population initialization intervals [-40,40], [-60,60], and [-100,100]; **D=1000** and $\mathbf{N_P}$=**{6,100}**.

F	N_P=6					N_P=100				
	[-100,100]	[-60,60]		[-40,40]		[-100,100]	[-60,60]		[-40,40]	
	Error	Error	W	Error	W	Error	Error	W	Error	W
1	1.59e+06±1.04e+05	5.82e+05±4.05e+04	−	2.58e+05±1.80e+04	−	1.11e+06±7.31e+04	4.02e+05±2.48e+04	−	1.79e+05±1.23e+04	−
2	9.42e+10±1.16e+10	3.35e+10±4.39e+09	−	1.51e+10±1.46e+09	−	3.05e+10±3.13e+09	1.13e+10±9.98e+08	−	4.99e+09±4.38e+08	−
3	3.41e+11±3.13e+11	1.57e+09±9.18e+08	−	8.32e+07±3.49e+07	−	5.22e+10±4.67e+10	3.35e+08±1.70e+08	−	1.68e+07±5.32e+06	−
4	2.62e+08±2.17e+07	8.50e+07±7.18e+06	−	3.44e+07±2.97e+06	−	1.67e+08±1.11e+07	5.33e+07±3.26e+06	−	2.00e+07±1.66e+06	−
5	6.84e+04±4.28e+03	4.42e+04±1.80e+03	−	3.28e+04±1.60e+03	−	3.08e+04±1.64e+03	2.06e+04±1.15e+03	−	1.55e+04±7.81e+02	−
6	2.73e+09±1.61e+08	9.84e+08±9.23e+07	−	4.98e+08±4.50e+07	−	1.39e+09±1.35e+08	5.42e+08±4.18e+07	−	2.78e+08±2.23e+07	−
7	1.36e+07±1.05e+06	4.80e+06±4.37e+05	−	2.17e+06±1.73e+05	−	8.19e+06±6.13e+05	2.81e+06±1.79e+05	−	1.27e+06±8.57e+04	−
8	1.27e+14±1.69e+13	1.67e+13±1.68e+12	−	3.25e+12±3.88e+11	−	7.85e+13±1.06e+13	9.99e+12±1.08e+12	−	1.98e+12±2.90e+11	−
9	1.74e+14±2.62e+13	2.24e+13±4.07e+12	−	4.50e+12±6.35e+11	−	8.78e+13±1.12e+13	1.20e+13±1.38e+12	−	2.27e+12±2.62e+11	−
10	8.09e+10±9.13e+09	2.93e+10±3.81e+09	−	1.32e+10±1.83e+09	−	4.08e+10±4.09e+09	1.47e+10±1.70e+09	−	6.93e+09±7.87e+08	−
11	3.69e+06±6.56e+05	1.41e+06±3.60e+05	−	5.74e+05±1.44e+05	−	3.31e+06±4.26e+05	1.13e+06±1.21e+05	−	5.17e+05±5.43e+04	−
12	2.61e+25±7.49e+25	8.62e+19±8.11e+19	=	1.74e+17±3.95e+17	=	5.34e+26±8.76e+26	2.79e+21±4.64e+21	−	5.78e+17±5.32e+17	−
13	2.51e+05±1.13e+04	1.51e+05±6.95e+03	−	1.00e+05±4.38e+03	−	2.07e+05±5.86e+03	1.23e+05±3.60e+03	−	8.32e+04±2.15e+03	−
14	9.09e+05±1.37e+05	2.33e+05±3.98e+04	−	7.75e+04±1.18e+04	−	6.77e+05±1.16e+05	1.54e+05±3.15e+04	−	5.12e+04±1.02e+04	−
15	1.66e+11±1.66e+11	8.43e+08±5.15e+08	−	6.35e+07±2.96e+07	−	2.32e+11±2.19e+11	9.62e+08±6.38e+08	−	3.35e+07±1.22e+07	−
16	1.30e+04±1.20e+03	4.02e+03±3.28e+02	−	1.52e+03±1.36e+02	−	8.81e+03±6.08e+02	2.72e+03±1.96e+02	−	1.02e+03±5.75e+01	−
17	3.34e+08±4.48e+08	8.69e+06±4.32e+06	−	2.09e+06±4.12e+05	−	7.60e+09±7.69e+09	1.29e+07±3.23e+06	−	1.80e+06±2.10e+05	−
18	2.03e+09±1.72e+09	1.24e+07±4.57e+06	−	2.63e+06±4.48e+05	−	1.57e+10±2.17e+10	3.54e+07±2.05e+07	−	2.63e+06±5.49e+05	−
19	4.40e+08±6.98e+07	5.87e+07±9.91e+06	−	1.15e+07±1.92e+06	−	2.24e+08±3.47e+07	2.86e+07±3.38e+06	−	5.60e+06±5.25e+05	−
20	2.40e+09±1.43e+08	8.83e+08±6.91e+07	−	3.54e+08±2.21e+07	−	1.64e+09±1.15e+08	5.58e+08±3.54e+07	−	2.33e+08±1.72e+07	−
21	1.29e+06±1.19e+05	4.01e+05±3.05e+04	−	1.47e+05±1.38e+04	−	8.70e+05±3.65e+04	2.63e+05±1.29e+04	−	9.53e+04±6.37e+03	−
22	1.29e+06±1.09e+05	3.90e+05±2.54e+04	−	1.43e+05±1.15e+04	−	8.89e+05±6.51e+04	2.72e+05±1.74e+04	−	9.32e+04±6.72e+03	−
23	4.41e+35±1.43e+35	6.35e+24±2.33e+34	=	8.24e+35±3.63e+35	=	9.21e+35±1.53e+34	2.65e+35±4.33e+34	=	1.11e+35±3.43e+34	=
24	1.25e+10±1.08e+09	3.85e+09±3.15e+08	−	1.44e+09±1.13e+08	−	8.87e+09±6.97e+08	2.60e+09±2.20e+08	−	9.37e+08±6.34e+07	−

TABLE IV: Summary of Wilcoxon statistical test for population initialization interval [-100,100] versus [-60,60] and [-40,40], with $N_P = \{6, 100\}$, and for $D = \{10, 100, 1000\}$.

N_P	6						100					
Interval	[-60,60]			[-40,40]			[-60,60]			[-40,40]		
W / D	−	=	+	−	=	+	−	=	+	−	=	+
10	21	3	0	21	3	0	7	17	0	10	14	0
100	23	1	0	23	1	0	22	2	0	22	2	0
1000	22	2	0	22	2	0	23	1	0	23	1	0

Polytechnica Hungarica, vol. 6, no. 5, pp. 95–115, 2009.

[15] Y. Xu, L. Wang, and L. Li, "An effective hybrid algorithm based on simplex search and differential evolution for global optimization," in *Emerging Intelligent Computing Technology and Applications. With Aspects of Artificial Intelligence*. Springer, 2009, pp. 341–350.

[16] R. A. Khanum and M. A. Jan, "Centroid-based initialized jade for global optimization," in *Computer Science and Electronic Engineering Conference (CEEC), 2011 3rd*. IEEE, 2011, pp. 115–120.

[17] I. H. Sloan and S. Joe, *Lattice methods for multiple integration*. Oxford University Press, 1994.

[18] H. Maaranen, K. Miettinen, and M. M. Mäkelä, "Quasi-random initial population for genetic algorithms," *Computers & Mathematics with Applications*, vol. 47, no. 12, pp. 1885–1895, 2004.

[19] A. Esmailzadeh and S. Rahnamayan, "Center-point-based simulated annealing," in *Electrical & Computer Engineering (CCECE), 2012 25th IEEE Canadian Conference on*. IEEE, 2012, pp. 1–4.

[20] W. J. Morokoff and R. E. Caflisch, "Quasi-random sequences and their discrepancies," *SIAM Journal on Scientific Computing*, vol. 15, no. 6, pp. 1251–1279, 1994.

[21] M. Pant, M. Ali, and V. Singh, "Differential evolution using quadratic interpolation for initializing the population," in *Advance Computing Conference, 2009. IACC 2009. IEEE International*. IEEE, 2009, pp. 375–380.

[22] K. Parsopoulos and M. Vrahatis, "Initializing the particle swarm optimizer using the nonlinear simplex method," *Advances in intelligent systems, fuzzy systems, evolutionary computation*, vol. 216, pp. 1–6, 2002.

[23] M. Ali, M. Pant, and A. Abraham, "Unconventional initialization methods for differential evolution," *Applied Mathematics and Computation*, vol. 219, no. 9, pp. 4474–4494, 2013.

[24] V. V. de Melo and A. C. B. Delbem, "Investigating smart sampling as a population initialization method for differential evolution in continuous problems," *Information Sciences*, vol. 193, pp. 36–53, 2012.

[25] G. Zhang, L. Gao, and Y. Shi, "An effective genetic algorithm for the flexible job-shop scheduling problem," *Expert Systems with Applications*, vol. 38, no. 4, pp. 3563–3573, 2011.

[26] R. Kumar, S. Narula, and R. Kumar, "A population initialization method by memetic algorithm," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 4, 2013.

[27] H. Salehinejad, S. Rahnamayan, and H. R. Tizhoosh, "Type-ii opposition-based differential evolution," in *2014 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2014, pp. 1768–1775.

[28] S. Rahnamayan, H. R. Tizhoosh, and M. M. Salama, "Opposition versus randomness in soft computing techniques," *Applied Soft Computing*, vol. 8, no. 2, pp. 906–918, 2008.

[29] S. Rahnamayan, G. G. Wang, and M. Ventresca, "An intuitive distance-based explanation of opposition-based sampling," *Applied Soft Computing*, vol. 12, no. 9, pp. 2828–2839, 2012.

[30] S. Rahnamayan, H. R. Tizhoosh, and M. Salama, "Quasi-oppositional differential evolution," in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*. IEEE, 2007, pp. 2229–2236.

[31] A. Esmailzadeh and S. Rahnamayan, "Enhanced differential evolution using center-based sampling," in *Evolutionary Computation (CEC), 2011 IEEE Congress on*. IEEE, 2011, pp. 2641–2648.

[32] S. Rahnamayan, J. Jesuthasan, F. Bourennani, G. F. Naterer, and H. Salehinejad, "Centroid opposition-based differential evolution," *International Journal of Applied Metaheuristic Computing (IJAMC)*, vol. 5, no. 4, pp. 1–25, 2014.

[33] S. Rahnamayan, J. Jesuthasan, F. Bourennani, H. Salehinejad, and G. F. Naterer, "Computing opposition by involving entire population," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 1800–1807.

[34] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *Evolutionary Computation, IEEE Transactions on*, vol. 15, no. 1, pp. 4–31, 2011.

[35] S. Rahnamayan and G. G. Wang, "Center-based sampling for population-based algorithms," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*. IEEE, 2009, pp. 933–938.

[36] N. Hansen, S. Finck, R. Ros, and A. Auger, "Real-parameter black-box optimization benchmarking 2010: Noiseless functions definitions," 2010-compiled 2014.