# On Tenfold Execution Time in Real World Optimization Problems With Differential Evolution in Perspective of Algorithm Design

Aleš Zamuda, *Senior Member, IEEE* and Janez Brest, *Senior Member, IEEE*
University of Maribor, Faculty of Electrical Engineering and Computer Science
Koroška cesta 46, 2000 Maribor, Slovenia
*ales.zamuda@um.si*

***Abstract*—This paper presents an algorithm design perspective on tenfold execution time in Real World Industry Challenges (RWIC) optimized using Differential Evolution (DE). The DE is a branch of optimization algorithms that include a set of design decisions to make before committing a finally proposed DE algorithm for a specific RWIC at hand. Therefore, analysis of a well known DE example variant algorithm on RWIC benchmark is reported, based on Competition on Testing Evolutionary Algorithms on Real World Optimization at Congress on Evolutionary Computation (CEC) 2011. The number of fitness functions allowed to execute the optimization runs is reduced or expanded, i.e. tenfolded in ten times of execution limit, and then the corresponding performance of the respective algorithms is compared to performance with the original execution limit. Discussion on aggregated performance with this runtime is then provided in the perspective of new algorithm design.**

***Index Terms*—Differential Evolution; Real World Optimization; Performance Evaluation; Benchmarking; Runtime**

## I. Introduction

A perspective on algorithm design using tenfold execution time in Real World Industry Challenges (RWIC) [1] is addressed in this paper. The optimization is conducted using Differential Evolution (DE) [2]. The DE is a branch of optimization algorithms that include a set of design decisions to make before committing a finally proposed DE algorithm for a specific RWIC at hand. Therefore, analysis of a well known DE example variant algorithm on RWIC benchmark is reported, based on Competition on Testing Evolutionary Algorithms on Real World Optimization at Congress on Evolutionary Computation (CEC) 2011 [1]. The number of fitness functions allowed to execute the optimization runs is reduced or expanded in tenfolded and then the corresponding performance of the respective algorithms is compared to performance with the original execution limit. The motivation to observe this is in the light of assessing the progress of an evolutionary algorithm development, similar to [3]. The aggregated performance is measured relatively to the algorithms from the CEC 2011 competition and it is shown how much the ranks change when tenfolding execution time for the example algorithm. Discussion on performance with this runtime is then provided in the perspective of new algorithm design. Specifically, the rankings are observed when changing the runtime, relatively to the CEC 2011 competition algorithms. In this paper, we are going to observe tenfolding as a measure of magnitudes in execution runtime for number of fitness evaluations limit in optimization.

Next section presents related work on DE and the utilized DE example algorithm in performance evaluation reports. Section 3 presents the architecture of tenfold execution time perspective. Section 4 reports the experiments results and discussion on ranking the performance in the proposed perspective. Section 5 conveys the conclusions and suggestions for future work.

## II. Related Work

Differential Evolution (DE) was proposed by Storn and Price [4] and since then, it has been modified and extended several times with new modifications and enhancements proposed [5], [6], [7], [8], [9], [10], [11], [12]. In general, DE is a floating-point encoding optimization algorithm that can be applied for global optimization over continuous or discrete function spaces [13], [14], [25], [15], [16], [17]. DE has many application domains, from multi-objective optimization [16], [18] to real world problems tacked [19], [20], [21], [22]. Advances, success, and comprehensive survey on DE are found in papers like [23], [24], [2], [3], which demonstrate the vast applicability of DE in recent decades.

The DE algorithm [4] has a main evolution loop counting the current generation number $g$, where a population of size $NP$ consists of vectors computed in this loop. For each vector $\mathbf{x}_i$, $\forall i \in \{1, 2, ..., NP\}$ in the current population during each generation $g$, DE employs following evolutionary operators. Namely, these operators are mutation, crossover, and selection. With these operators, DE produces a trial vector (offspring) and selects one of the vectors with the best fitness value.

The mutation operator creates a mutant vector $\mathbf{v}_{i,g+1}$ for each corresponding population vector. One of the most popular DE mutation operators among many proposed in other works are *'rand/1'* [13], [4]:

$$\mathbf{v}_{i,g+1} = \mathbf{x}_{r_1,g} + F(\mathbf{x}_{r_2,g} - \mathbf{x}_{r_3,g}) \tag{1}$$

and *'best/1'*:

$$\mathbf{v}_{i,g+1} = \mathbf{x}_{best,g} + F(\mathbf{x}_{r_1,g} - \mathbf{x}_{r_2,g}), \tag{2}$$

where the indexes $r_1$, $r_2$, and $r_3$ represent the randomly chosen and mutually different integers, generated within the range $\{1, NP\}$ and also different from index $i$ for each vector. Currently best vector is denoted as $\mathbf{x}_{best,g}$. $F$ is an amplification factor of the difference vector, usually within the interval $[0, 1]$. Vector at index $r_1$ is a base vector. The term $\mathbf{x}_{r_2,g} - \mathbf{x}_{r_3,g}$ denotes a difference vector, which after multiplication with $F$, is named amplified difference vector.

The crossover operation follows the mutation operation. Here, the mutant vector $\mathbf{v}_{i,g+1}$ is taken into recombination process with the target vector $\mathbf{x}_{i,g}$, to create a trial vector $u_{i,j,g+1}$. The binary crossover operator is defined as:

$$u_{i,j,G+1} = \begin{cases} v_{i,j,G+1} & \text{if } rand(0,1) \leq CR \text{ or } j = j_{\text{rand}} \\ x_{i,j,G} & \text{otherwise} \end{cases}, \tag{3}$$

where $j \in \{1, 2, ..., D\}$ denotes the $j$-th search parameter of $D$-dimensional search space and $rand(0, 1) \in [0, 1]$ denotes a uniformly distributed random number. $j_{\text{rand}}$ denotes a uniform randomly chosen index of the search parameter that is always exchanged to prevent cloning of target vectors. $CR$ denotes the crossover rate and the setting and variation of these values have been extensively studied in the literature [26], [27].

The selection operator as the last operator to compute one DE generation, propagates the fittest individual in the new generation (for minimization problem):

$$\mathbf{x}_{i,G+1} = \begin{cases} \mathbf{u}_{i,G+1} & \text{if } f(\mathbf{u}_{i,G+1}) < f(\mathbf{x}_{i,G}) \\ \mathbf{x}_{i,G} & \text{otherwise} \end{cases}. \tag{4}$$

To change the control parameters $F$ and $CR$ during the evolutionary process, the jDE algorithm [5] extends the original DE algorithm with a self-adaptive control mechanism. The third control parameter $NP$ is kept unchanged in [5]. Each individual in the jDE population is extended using the values of these two control parameters. New control parameters $F_{i,g+1}$ and $CR_{i,g+1}$ are calculated as [5]: They produce control parameters $F$ and $CR$ in a new vector. The $rand_j \in [0, 1]$, $j \in \{1, 2, 3, 4\}$ are uniform random values. $\tau_1$ and $\tau_2$ represent the probabilities of adjusting control parameters $F$ and $CR$, respectively. $\tau_1, \tau_2, F_l, F_u$ are taken fixed values as proposed in [5], but they can be changed as suggested in [27]. The new $F$ or $CR$ are assigned a random value, within $[0.1, 1.0]$ or within $[0, 1]$, respectively. Both $F_{i,g+1}$ and $CR_{i,g+1}$ are computed before the mutation operation is performed. So the new $F$ and $CR$ influence the mutation, crossover, and selection operations of the new vector $\mathbf{x}_{i,g+1}$.

In [23] it was reported that one of best extensions for jDE is dynNP-DE [28]. With this extension, DE reduces population size by half, when certain generations are reached. These reductions are applied, when the current generation number $g_p$ is greater than the ratio between the total number of function evaluations allowed $N_{\text{max\_Feval}}$ and the population size in the current generation $NP_p$:

$$g_p > \frac{N_{\text{max\_Feval}}}{p_{\max} NP_p},$$

To discard half of vectors from a larger population, vectors are pairwise compared to their corresponding index neighbor, as defined and drawn in [28].

## III. Tenfold Execution Time Perspective

The algorithm dynNP-DE [28] has not yet been applied to CEC 2011, specifically in the context of tenfold execution time. Recently, an interesting study regarding the stepwise improvement of DE algorithms has been published [3], applying the perspective of multiplication for maximum number of fitness functions allowed. The perspective of increasing or decreasing of optimization runtime to an order of magnitude is however, most important in the context of real world optimization. One such example is piloting missions of underwater ocean autonomous vehicles [29], which communicate through limited expensive satellite connection. These missions are highly unpredictable and the assessment of runtime for optimization algorithms is of high value here, which further motivates observance of significant performance differences in results yielded from optimization algorithms. In this paper, we observe tenfolding as a measure of magnitudes in execution runtime for number of fitness evaluations in optimization.

## IV. Results

To study the performance in the perspective of tenfolding execution time for real world optimization in this paper, the benchmark from Competition on Testing Evolutionary Algorithms on Real World Optimization at Congress on Evolutionary Computation (CEC) 2011 [1] is used.

Using dynNP-DE [28] algorithm, in Table IV results are first reported for 150000 fitness evaluations and then for tenfolded fitness evaluations, i.e. 15000 (reduced execution time) and 1500000 (extended execution time). For dynNP-DE, initial population size ($NP$) is set at 200 and number of reductions at $pmax = 4$. As can be seen from Table IV based on the results from Table IV, the results improve several times for dynNP-DE when changing execution time, however, it is not easy to judge how much the improvement should be valued.

Therefore, in Table IV, the average results from Table IV for 15000, 150000, and 1500000 fitness evaluations at each function are ranked. The ranking is done against the 14 entry algorithms from CEC 2011 competition, based on the aggregated results from the competition website file[1]. The changes in the last colum of Table IV show that there is 23 improvements when expanding the number of fitness evaluations tenfold. Also, one more best average result can be obtained with tenfolding number of fitness evaluations.

## V. Conclusion

In this paper, tenfolding of number of fitness evaluations was evaluated in the perspective of optimization runtime, which is important in real world optimization challenges, especially when designing a new algorithm or applying an algorithm in the new or changing environment.

[1] http://www3.ntu.edu.sg/home/epnsugan/index_files/CEC11-RWP/ranking.xlsx

TABLE I

BEST, WORST, MEDIAN, AVERAGE VALUES AND THEIR STANDARD DEVIATION FOR 25 INDEPENDENT RUN RESULTS ON SELECTED CEC 2011 FUNCTIONS USING ALGORITHM DYNNP-DE [28], FOR MAXIMUM FUNCTION EVALUATIONS OF 15000, 150000, AND 1500000.

| Fun. | FES | Best | Worst | Median | Average | Std. dev. |
|---|---|---|---|---|---|---|
| F1 | 15000 | 7.6866e-02 | 2.0077e+01 | 1.2440e+01 | 1.0718e+01 | 6.6950e+00 |
| F2 | 15000 | -1.7403e+01 | -9.9315e+00 | -1.3360e+01 | -1.3329e+01 | 2.1537e+00 |
| F3 | 15000 | 1.1515e-05 | 1.1515e-05 | 1.1515e-05 | 1.1515e-05 | 2.4667e-19 |
| F4 | 15000 | 1.3771e+01 | 2.0957e+01 | 1.5325e+01 | 1.7240e+01 | 3.3297e+00 |
| F5 | 15000 | -3.3586e+01 | -2.6434e+01 | -3.0900e+01 | -3.0490e+01 | 1.9847e+00 |
| F6 | 15000 | -2.7427e+01 | -2.0465e+01 | -2.2600e+01 | -2.3421e+01 | 2.2169e+00 |
| F7 | 15000 | 1.4525e+00 | 1.9691e+00 | 1.6970e+00 | 1.6694e+00 | 1.4323e-01 |
| F8 | 15000 | 2.2000e+02 | 2.2000e+02 | 2.2000e+02 | 2.2000e+02 | 0.0000e+00 |
| F9 | 15000 | 6.7241e+03 | 4.1628e+04 | 1.8858e+04 | 1.9908e+04 | 8.8894e+03 |
| F10 | 15000 | -1.9197e+01 | -1.2669e+01 | -1.4903e+01 | -1.4971e+01 | 1.9003e+00 |
| F11 | 15000 | 6.5166e+05 | 1.5780e+06 | 1.1531e+06 | 1.1040e+06 | 2.5035e+05 |
| F12 | 15000 | 1.1947e+06 | 2.2850e+06 | 1.5549e+06 | 1.6312e+06 | 2.8296e+05 |
| F13 | 15000 | 1.5445e+04 | 1.5493e+04 | 1.5459e+04 | 1.5464e+04 | 1.5471e+01 |
| F14 | 15000 | 1.8654e+04 | 1.9384e+04 | 1.8950e+04 | 1.8939e+04 | 1.7821e+02 |
| F15 | 15000 | 3.2781e+04 | 3.3145e+04 | 3.2962e+04 | 3.2968e+04 | 8.0419e+01 |
| F16 | 15000 | 1.3611e+05 | 1.5781e+05 | 1.4441e+05 | 1.4427e+05 | 5.9501e+03 |
| F17 | 15000 | 1.9296e+06 | 3.5391e+06 | 2.3767e+06 | 2.5059e+06 | 5.2167e+05 |
| F18 | 15000 | 9.4170e+05 | 1.5890e+06 | 9.7775e+05 | 1.0139e+06 | 1.3061e+05 |
| F19 | 15000 | 9.4170e+05 | 1.5890e+06 | 9.7775e+05 | 1.0139e+06 | 1.3061e+05 |
| F20 | 15000 | 9.4170e+05 | 1.5890e+06 | 9.7775e+05 | 1.0139e+06 | 1.3061e+05 |
| F21 | 15000 | 1.7625e+01 | 2.7563e+01 | 2.2576e+01 | 2.2753e+01 | 2.3430e+00 |
| F22 | 15000 | 1.4583e+01 | 2.7973e+01 | 2.0027e+01 | 2.0228e+01 | 3.3872e+00 |
| F1 | 150000 | 0.0000e+00 | 1.3127e-11 | 1.0854e-27 | 6.0448e-13 | 2.6388e-12 |
| F2 | 150000 | -2.5661e+01 | -2.1535e+01 | -2.3671e+01 | -2.3683e+01 | 1.0421e+00 |
| F3 | 150000 | 1.1515e-05 | 1.1515e-05 | 1.1515e-05 | 1.1515e-05 | 2.0039e-19 |
| F4 | 150000 | 1.3771e+01 | 2.0957e+01 | 1.4329e+01 | 1.6329e+01 | 3.2171e+00 |
| F5 | 150000 | -3.6720e+01 | -3.3926e+01 | -3.5043e+01 | -3.5097e+01 | 9.8899e-01 |
| F6 | 150000 | -2.9166e+01 | -2.6558e+01 | -2.9008e+01 | -2.8559e+01 | 8.3283e-01 |
| F7 | 150000 | 1.0152e+00 | 1.4320e+00 | 1.2687e+00 | 1.2481e+00 | 1.2710e-01 |
| F8 | 150000 | 2.2000e+02 | 2.2000e+02 | 2.2000e+02 | 2.2000e+02 | 0.0000e+00 |
| F9 | 150000 | 1.4639e+03 | 3.1297e+03 | 2.1494e+03 | 2.1449e+03 | 4.4661e+02 |
| F10 | 150000 | -2.1570e+01 | -2.1209e+01 | -2.1357e+01 | -2.1367e+01 | 6.9669e-02 |
| F11 | 150000 | 5.1021e+04 | 7.2823e+04 | 5.2263e+04 | 5.3178e+04 | 4.1664e+03 |
| F12 | 150000 | 1.0699e+06 | 1.0791e+06 | 1.0750e+06 | 1.0747e+06 | 2.2129e+03 |
| F13 | 150000 | 1.5444e+04 | 1.5449e+04 | 1.5444e+04 | 1.5445e+04 | 1.5964e+00 |
| F14 | 150000 | 1.8207e+04 | 1.8692e+04 | 1.8465e+04 | 1.8476e+04 | 9.6231e+01 |
| F15 | 150000 | 3.2755e+04 | 3.2932e+04 | 3.2854e+04 | 3.2848e+04 | 5.5660e+01 |
| F16 | 150000 | 1.3048e+05 | 1.4435e+05 | 1.3310e+05 | 1.3386e+05 | 2.8150e+03 |
| F17 | 150000 | 1.9221e+06 | 2.2130e+06 | 1.9591e+06 | 1.9905e+06 | 7.4544e+04 |
| F18 | 150000 | 9.4096e+05 | 9.5191e+05 | 9.4409e+05 | 9.4429e+05 | 2.5702e+03 |
| F19 | 150000 | 1.0100e+06 | 1.3440e+06 | 1.1827e+06 | 1.1810e+06 | 9.5643e+04 |
| F20 | 150000 | 9.4096e+05 | 9.5191e+05 | 9.4409e+05 | 9.4429e+05 | 2.5702e+03 |
| F21 | 150000 | 9.7205e+00 | 1.9251e+01 | 1.6850e+01 | 1.6618e+01 | 1.9838e+00 |
| F22 | 150000 | 9.0622e+00 | 1.6330e+01 | 1.3381e+01 | 1.3107e+01 | 1.8853e+00 |
| F1 | 1500000 | 0.0000e+00 | 0.0000e+00 | 0.0000e+00 | 0.0000e+00 | 0.0000e+00 |
| F2 | 1500000 | -2.8423e+01 | -2.5423e+01 | -2.7202e+01 | -2.7199e+01 | 7.7960e-01 |
| F3 | 1500000 | 1.15149e-05 | 1.15149e-05 | 1.15149e-05 | 1.15149e-05 | 2.0039e-19 |
| F4 | 1500000 | 1.3771e+01 | 2.0957e+01 | 1.4329e+01 | 1.6329e+01 | 3.2171e+00 |
| F5 | 1500000 | -3.6845e+01 | -3.5913e+01 | -3.6835e+01 | -3.6720e+01 | 2.9505e-01 |
| F6 | 1500000 | -2.9166e+01 | -2.9157e+01 | -2.9165e+01 | -2.9165e+01 | 2.0318e-03 |
| F7 | 1500000 | 5.0000e-01 | 1.1263e+00 | 9.7443e-01 | 9.4912e-01 | 1.5678e-01 |
| F8 | 1500000 | 2.2000e+02 | 2.2000e+02 | 2.2000e+02 | 2.2000e+02 | 0.0000e+00 |
| F9 | 1500000 | 1.3603e+03 | 2.6550e+03 | 1.9363e+03 | 1.9931e+03 | 3.4255e+02 |
| F10 | 1500000 | -2.1644e+01 | -2.1354e+01 | -2.1398e+01 | -2.1394e+01 | 5.8419e-02 |
| F11 | 1500000 | 5.1147e+04 | 5.3529e+04 | 5.2263e+04 | 5.2174e+04 | 5.5760e+02 |
| F12 | 1500000 | 1.0713e+06 | 1.0789e+06 | 1.0745e+06 | 1.0745e+06 | 1.8061e+03 |
| F13 | 1500000 | 1.5444e+04 | 1.5449e+04 | 1.5444e+04 | 1.5445e+04 | 1.6296e+00 |
| F14 | 1500000 | 1.8021e+04 | 1.8267e+04 | 1.8124e+04 | 1.8129e+04 | 5.9713e+01 |
| F15 | 1500000 | 3.2693e+04 | 3.2753e+04 | 3.2698e+04 | 3.2703e+04 | 1.4712e+01 |
| F16 | 1500000 | 1.2711e+05 | 1.2979e+05 | 1.2848e+05 | 1.2839e+05 | 8.1723e+02 |
| F17 | 1500000 | 1.8933e+06 | 1.9590e+06 | 1.9327e+06 | 1.9276e+06 | 1.8533e+04 |
| F18 | 1500000 | 9.3606e+05 | 9.4345e+05 | 9.4032e+05 | 9.4020e+05 | 1.7456e+03 |
| F19 | 1500000 | 9.4999e+05 | 1.1307e+06 | 1.0214e+06 | 1.0210e+06 | 5.2718e+04 |
| F20 | 1500000 | 9.3606e+05 | 9.4345e+05 | 9.4032e+05 | 9.4020e+05 | 1.7456e+03 |
| F21 | 1500000 | 1.1823e+01 | 1.6766e+01 | 1.4564e+01 | 1.4617e+01 | 1.2394e+00 |
| F22 | 1500000 | 8.6102e+00 | 1.5498e+01 | 1.0292e+01 | 1.1377e+01 | 2.5562e+00 |

TABLE II
AVERAGE RESULTS FOR DYNNP-DE AND THEIR CHANGE AT DIFFERENT TENFOLDING EXECUTION LIMIT.

| Function | 15000 FES | 150000 FES difference | 1500000 FES difference |
|---|---|---|---|
| F1 | 6.0448e-13 | 10.718 | -6.0448e-13 |
| F2 | -2.3683e+01 | 10.354 | -3.516 |
| F3 | 1.1515e-05 | 0 | -1e-10 |
| F4 | 1.6329e+01 | 0.911 | 0 |
| F5 | -3.5097e+01 | 4.607 | -1.623 |
| F6 | -2.8559e+01 | 5.138 | -0.606 |
| F7 | 1.2481e+00 | 0.4213 | -0.29898 |
| F8 | 2.2000e+02 | 0 | 0 |
| F9 | 2.1449e+03 | 17763.1 | -151.8 |
| F10 | -2.1367e+01 | 6.396 | -0.027 |
| F11 | 5.3178e+04 | 1050822 | -1004 |
| F12 | 1.0747e+06 | 556500 | -200 |
| F13 | 1.5445e+04 | 19 | 0 |
| F14 | 1.8476e+04 | 463 | -347 |
| F15 | 3.2848e+04 | 120 | -145 |
| F16 | 1.3386e+05 | 10410 | -5470 |
| F17 | 1.9905e+06 | 515400 | -62900 |
| F18 | 9.4429e+05 | 69610 | -4090 |
| F19 | 1.1810e+06 | -167100 | -160000 |
| F20 | 9.4429e+05 | 69610 | -4090 |
| F21 | 1.6618e+01 | 6.135 | -2.001 |
| F22 | 1.3107e+01 | 7.121 | -1.73 |

We ranked the performance changes of a well known DE algorithm against the competing algorithms from CEC 2011 competition at 150000 fitness evaluations, by changing the number of fitness evaluations for this algorithm. The number of fitness evaluations increased or decreased was for ten, which is less than the count of all competition entry algorithms. We observed that when expanding the number of fitness evaluations, the yielded results of the algorithm can reach a first rank in one of functions.

In the future work, impact of changing dimension and complexity of benchmark problems could be addressed within the proposed perspective. Also, the application of different execution environments, like the high-performance computing equipment, and cost of real world problems simulations and design, could be studied. Also, the impact of parameters and hyper-heuristic approaches to algorithm design might be taken in account. Similar analysis of other algorithms could also be reported in this perspective.

## ACKNOWLEDGMENT

The authors acknowledge the financial support from the Slovenian Research Agency (research core funding No. P2-0041). This paper is also based upon work from COST Action CA15140 'Improving Applicability of Nature-Inspired Optimisation by Joining Theory and Practice (ImAppNIO) and COST Action IC1406 'High-Performance Modelling and Simulation for Big Data Applications (cHiPSet)' supported by COST (European Cooperation in Science and Technology).

## REFERENCES

[1] S. Das and P. N. Suganthan, "Problem Definitions and Evaluation Criteria for CEC 2011 Competition on Testing Evolutionary Algorithms on Real World Optimization Problems," Jadavpur University, India & Nanyang Technological University, Technical Report, 2011.

[2] S. Das, S. S. Mullick, and P. Suganthan, "Recent advances in differential evolution – An updated survey," *Swarm and Evolutionary Computation*, vol. 27, pp. 1–30, 2016.

[3] A. P. Piotrowski and J. J. Napiorkowski, "Step-by-step improvement of JADE and SHADE-based algorithms: Success or failure?" *Swarm and Evolutionary Computation*, 2018, DOI: https://doi.org/10.1016/j.swevo.2018.03.007.

[4] R. Storn and K. Price, "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.

[5] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer, "Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[6] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[7] J. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[8] S. Das, A. Abraham, U. Chakraborty, and A. Konar, "Differential Evolution Using a Neighborhood-based Mutation Operator," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 526–553, 2009.

[9] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66, 2011.

[10] E. Mininno, F. Neri, F. Cupertino, and D. Naso, "Compact Differential Evolution," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 32–54, 2011.

TABLE III
THE EXPERIMENT RESULTS WITH TENFOLDING FES FOR ALGORITHM DYNNP-DE, RANKED AGAINST OTHER ALGORITHMS FROM CEC 2011 COMPETITION, BASED ON THEIR MEAN VALUES.

| Function | 15000 FES | 150000 FES | 1500000 FES | Tenfold reduction (/ 10) difference | Tenfold expansion ($\times$10) difference |
|---|---|---|---|---|---|
| F1 | 9 | 5 | 1 | -4 | -4 |
| F2 | 8 | 6 | 6 | -2 | -2 |
| F3 | 2 | 2 | 2 | 0 | 0 |
| F4 | 8 | 8 | 8 | 0 | 0 |
| F5 | 12 | 10 | 10 | -2 | -2 |
| F6 | 4 | 3 | 3 | -1 | -1 |
| F7 | 10 | 9 | 8 | -1 | -1 |
| F8 | 1 | 1 | 1 | 0 | 0 |
| F9 | 11 | 10 | 10 | -1 | -1 |
| F10 | 14 | 10 | 10 | -4 | -4 |
| F11 | 15 | 12 | 12 | -3 | -3 |
| F12 | 7 | 4 | 4 | -3 | -3 |
| F13 | 8 | 3 | 3 | -5 | -5 |
| F14 | 7 | 5 | 3 | -2 | -2 |
| F15 | 7 | 6 | 3 | -1 | -1 |
| F16 | 11 | 9 | 4 | -2 | -2 |
| F17 | 14 | 13 | 12 | -1 | -1 |
| F18 | 13 | 9 | 6 | -4 | -4 |
| F19 | 8 | 10 | 8 | 2 | 2 |
| F20 | 12 | 7 | 7 | -5 | -5 |
| F21 | 15 | 14 | 12 | -1 | -1 |
| F22 | 15 | 12 | 12 | -3 | -3 |
| Sum | 211 | 168 | 145 | -43 | **23** |
| Worst count | 3 | 0 | 0 | -3 | 0 |
| Best count | 1 | 1 | 2 | 0 | 1 |


[11] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Applied Soft Computing*, vol. 11, no. 2, pp. 1679–1696, 2011.

[12] R. Tanabe and A. S. Fukunaga, "Improving the search performance of SHADE using linear population size reduction," in *2014 IEEE Congress on Evolutionary Computation*. IEEE, 2014, pp. 1658–1665.

[13] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, ser. Natural Computing Series. Berlin, Germany: Springer-Verlag, 2005.

[14] V. Feoktistov, *Differential Evolution: In Search of Solutions (Springer Optimization and Its Applications)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[15] A. Zamuda, J. Brest, B. Bošković, and V. Žumer, "Large Scale Global Optimization Using Differential Evolution with Self Adaptation and Cooperative Co-evolution," in *2008 IEEE World Congress on Computational Intelligence*. IEEE Press, 2008, pp. 3719–3726.

[16] ——, "Differential Evolution with Self-adaptation and Local Search for Constrained Multiobjective Optimization," in *IEEE Congress on Evolutionary Computation 2009*. IEEE Press, 2009, pp. 195–202.

[17] J. Tvrdík, "Adaptation in differential evolution: A numerical comparison," *Applied Soft Computing*, vol. 9, no. 3, pp. 1149–1155, 2009.

[18] T. Tušar, P. Korošec, G. Papa, B. Filipič, and J. Šilc, "A comparative study of stochastic optimization methods in electric motor design," *Applied Intelligence*, vol. 2, no. 27, pp. 101–111, 2007.

[19] A. Zamuda, J. Brest, B. Bošković, and V. Žumer, "Differential Evolution for Parameterized Procedural Woody Plant Models Reconstruction," *Applied Soft Computing*, vol. 11, no. 8, pp. 4904–4912, 2011.

[20] A. Glotić and A. Zamuda, "Short-term combined economic and emission hydrothermal optimization by surrogate differential evolution," *Applied Energy*, vol. 141, pp. 42–56, 1 March 2015.

[21] A. Zamuda and J. D. Hernández Sosa, "Underwater glider path planning and population size reduction in differential evolution," in *Computer Aided Systems Theory – EUROCAST 2015*, ser. Lecture Notes in Computer Science, R. Moreno-Daz, F. Pichler, and A. Quesada-Arencibia, Eds. Springer International Publishing, 2015, vol. 9520, pp. 853–860.

[22] A. Zamuda, J. D. H. Sosa, and L. Adler, "Constrained Differential Evolution Optimization for Underwater Glider Path Planning in Submesoscale Eddy Sampling," *Applied Soft Computing*, vol. 42, pp. 93–118, 2016.

[23] F. Neri and V. Tirronen, "Recent Advances in Differential Evolution: A Survey and Experimental Analysis," *Artificial Intelligence Review*, vol. 33, no. 1–2, pp. 61–106, 2010.

[24] S. Das and P. N. Suganthan, "Differential Evolution: A Survey of the State-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.

[25] E. Mezura-Montes and B. C. Lopez-Ramirez, "Comparing bio-inspired algorithms in constrained optimization problems," *The 2007 IEEE Congress on Evolutionary Computation*, pp. 662–669, 25-28 Sept. 2007.

[26] D. Zaharie, "Influence of crossover on the behavior of Differential Evolution Algorithms," *Applied Soft Computing*, vol. 9, no. 3, pp. 1126–1138, 2009.

[27] A. Zamuda and J. Brest, "Self-adaptive control parameters' randomization frequency and propagations in differential evolution," *Swarm and Evolutionary Computation*, vol. 25, pp. 72–99, 2015.

[28] J. Brest and M. S. Maučec, "Population Size Reduction for the Differential Evolution Algorithm," *Applied Intelligence*, vol. 29, no. 3, pp. 228–247, 2008.

[29] A. Zamuda, J. D. H. Sosa, and L. Adler, "Improving Constrained Glider Trajectories for Ocean Eddy Border Sampling within Extended Mission Planning Time," in *2016 IEEE Congress on Evolutionary Computation*, 2016, pp. 1727–1734.