Eigen Crossover in Cooperative Model of Evolutionary Algorithms Applied to CEC 2022 Single Objective Numerical Optimisation

Petr Bujok
Department of Informatics and Computers
Faculty of Science, University of Ostrava
petr.bujok@osu.cz
ORCID: 0000-0003-2956-1226

Patrik Kolenovsky
Department of Informatics and Computers
Faculty of Science, University of Ostrava
patrik.kolenovsky@osu.cz

***Abstract*—In this paper, a cooperative model of four well-performing evolutionary algorithms enhanced by Eigen crossover is proposed and applied to a set of problems CEC 2022. The four adaptive algorithms employed in this model are – Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES), Differential Evolution with Covariance Matrix Learning and Bimodal Distribution Parameter Setting (CoBiDE), an adaptive variant of jSO, and Differential Evolution With an Individual-Dependent Mechanism (IDE). For the higher efficiency of the cooperative model, a linear population-size reduction mechanism is employed. The model was introduced for CEC 2019. Here, Eigen crossover is applied for each cooperating algorithm. The provided results show that the proposed model of four Evolutionary Algorithms with Eigen crossover (EA4eig) is able to solve ten out of 24 optimisation problems. Moreover, comparing EA4eig with four state-of-the-art variants of adaptive Differential Evolution illustrates the superiority of the newly designed optimiser.**

***Index Terms*—Differential Evolution, Evolution Strategy, cooperative model, competition, experiments, Eigen crossover**

I. Introduction

A new variant of cooperative model of Evolutionary Algorithms (EAs) is proposed and applied on a set of single-objective problems. For the objective function, $f(\boldsymbol{x}),\ \boldsymbol{x} = (x_1,\ x_2,\ \ldots, x_D) \in \mathbb{R}^D$ and the search domain Ω constrained by bounds, a lower limit (a_j) and an upper limit (b_j), $\Omega = \prod_{j=1}^{D}[a_j,\ b_j]$, $a_j < b_j$, $j = 1,\ 2,\ \ldots, D$, the global minimum point $\boldsymbol{x}^*$ satisfying condition $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}),\ \forall \boldsymbol{x} \in \Omega$ is the solution of the problem.

The global optimisation problem is solved in many various fields of research, industry, and healthcare services. Therefore, it is necessary to develop reliable, efficient, and fast methods to achieve better results and minimise economical and ecological loss. Various optimisation methods provide better results only on part of problems (No-Free-Lunch theorem [1]). Based on this aspect, it is natural to use the cooperation of different optimisation methods to cope with various problems. Very efficient optimisation methods inspired by Darwin's theory are called Evolutionary Algorithms. In this paper, a new variant of the cooperative model of Evolutionary Algorithms is designed and evaluated on the new CEC 2022 benchmark set.

One of the first cooperative models based on the Genetic Algorithm (CCGA) was proposed by Potter and De Jong in 1994 [2]. In this study, for each design variable one sub-population is generated and evaluated independently on the remaining sub-populations. The results achieved on four standard optimisation problems show that the model of sub-populations provides substantially faster convergence.

Ma et al. extended the CCGA model in the theoretical study in 2019 [3]. The authors analytically discussed possibilities for how to allocate design variables to sub-populations. The reason is in dependency of design variables in statically allocated sub-populations. The authors mainly proposed random allocation of sub-populations to design variables.

In 2020, Cai et al. proposed a study of cooperation between Evolutionary Algorithms and constrained handling techniques [4]. A model employing a variant of Differential Evolution with two variants of constrained handling technique (used in two phases) is applied on 24 test problems TR2006.

In 2021, Falcón-Cordoba et al. proposed and applied an Island-based Multi-indicator Algorithm (IMIA) for multi-objective problems [5]. The proposed model employing small sub-populations was implemented in a parallel regime using the OpenMP library. Results of the comparison IMIA are comparable with five existing methods.

In 2021, Fouad et al. proposed a model of Dynamic group-based cooperative optimisation algorithm (DGCO) [6]. The proposed model distinguishes exploration and exploitation phases in 70% of individuals for exploration and 30% for exploitation. The results on 23 multi-objective problems compared with five existing Evolutionary Algorithms show that the proposed DCGO algorithm is highly comparable.

The motivation for the cooperative model is based on previous research in this field of interest. In 2012, a cooperative model of six state-of-the-art DE variants was introduced [7]. This model was 2018 applied to real-world problems CEC 2011, where it achieved very promising results [8]. In 2015, a hierarchical parallel model of EAs was designed and applied to the CEC 2015 benchmark set [9]. In 2018, a cooperative model of eight popular nature-inspired algorithms

was applied to 22 real-world problems [10]. In 2019, a cooperative model of four efficient EAs was proposed for the competition of CEC 2019 [11]. The model was subsequently applied to the set of real-world problems CEC 2011 [12].

The rest of the paper is organised as follows. After the Introduction, a set of employed Evolutionary Algorithms are briefly described in Section II. An idea of the cooperative model is presented in Section III. The settings of the experimental part are provided in Section IV and results with statistical comparison of the newly proposed model are discussed in Section V. Finally, an assessment of the performance of the proposed model is provided in Section VI.

II. Optimisation Algorithms Selected for the Proposed Cooperative Model

Various optimisation algorithms provide good results on part of problems. There are a lot of efficient Evolutionary Algorithms, namely Differential Evolution (DE), Evolution Strategy (ES), Genetic Algorithm (GA), and others. The proposed cooperative model of Evolutionary Algorithms uses four different optimisation methods. Three of the EAs are successful adaptive DE and one is an efficient version of ES. The selection is based on very good results of the algorithms in various problems in previous experiments.

A. CoBiDE

In 2014, Wang et al. proposed the DE variant of CoBiDE, which employs bimodal distribution of the control parameters and covariance-matrix learning approach used for Eigen transformation [13]. A variant of CoBiDE brings to a classic DE two new aspects. The bimodal distribution setting of both F and *CR* parameters enables distinguishing between exploration and exploitation. Covariance-matrix learning for the crossover of the individuals enables rotating the coordinate system in order to adapt the dependencies in the population. It significantly increases performance when solving the rotated objective functions. A variant of this approach was also used in very efficient SPS-L-SHADE-EIG, which took the first position in the CEC 2015 competition [14].

For the initial values of F_i and CR_i, $i = 1, 2, \ldots N$ (N denotes population size) a Cauchy distribution is used:

$$F_i = \begin{cases} \text{randc}(0.65,\ 0.1) & \text{if } \text{rand}(0,\ 1) < 0.5 \\ \text{randc}(1.0,\ 0.1) & \text{otherwise,} \end{cases} \tag{1}$$

$$CR_i = \begin{cases} \text{randc}(0.1,\ 0.1) & \text{if } \text{rand}(0,\ 1) < 0.5 \\ \text{randc}(0.95,\ 0.1) & \text{otherwise.} \end{cases} \tag{2}$$

A mutation rand/1 is used to produce new solutions, using F_i value. Usage of the covariance-matrix based crossover is selected for a generation with a probability *pb*. In other cases, the classic binomial crossover is applied. The values of F_i and CR_i are regenerated if the new solution is not better than the parent solution. After generation, Eigenvalues (matrix $\boldsymbol{D}$) and Eigenvectors (matrix $\boldsymbol{B}$) are computed from the covariance matrix ($\boldsymbol{C}$) of a *ps* part of a better individual from population:

$$\boldsymbol{C} = \boldsymbol{B}\boldsymbol{D}^2\boldsymbol{B}^T. \tag{3}$$

A new solution is designed in an Eigen coordinate system:

$$\boldsymbol{x}_i^{'} = \boldsymbol{B}^{-1}\boldsymbol{x}_i = \boldsymbol{B}^T\boldsymbol{x}_i, \tag{4}$$

$$\boldsymbol{u}_i^{'} = \boldsymbol{B}^{-1}\boldsymbol{u}_i = \boldsymbol{B}^T\boldsymbol{u}_i. \tag{5}$$

Then, a binomial crossover is produces a new solution $\boldsymbol{y}_i^{'}$ which is transformed back into a standard coordinate system:

$$\boldsymbol{y}_i = \boldsymbol{B}\boldsymbol{y}_i^{'}. \tag{6}$$

B. IDEbd

In 2015, Tang et al. proposed a variant of DE with an Individual-Dependent approach (IDE) [15]. The search process is divided into explorative and exploitative stages. All individuals in the population are ordered in an ascending way with respect to the objective function value. The values of the parameters F and *CR* are for each individual computed using:

$$F_o = \frac{o}{N}, \tag{7}$$

$$CR_i = \frac{i}{N}, \tag{8}$$

where o is an index of a base individual used in mutation, i is an index of a current parent point, and N is a size of the population. Computed parameters F and *CR* are subsequently modified using normal distribution with variance 0.1, until they are in the interval $(0,\ 1)$.

The sorted individuals are divided into two sets - superior S (smaller cost function) and inferior I (higher cost function). The value of ps is the proportion of superior individuals in the population, and it is updated accordingly to:

$$ps = 0.1 + 0.9 \times 10^{5\times(g/g_{\max}-1)}, \tag{9}$$

where g is the current generation, and $g_{\max}$ is the maximum number of generations.

The original IDE algorithm uses a two-phase mutation based on the current base individual at the exploration phase and the randomly selected base vector at the second phase. This mutation controls convergence speed during the search. In 2017, the original IDE was enhanced by more progressive mutation variant (IDEbd) [16]:

$$\boldsymbol{u}_i = \begin{cases} \boldsymbol{x}_o + F_o * (\boldsymbol{x}_{r_1} - \boldsymbol{x}_o) + F_o * (\boldsymbol{x}_{r_2} - \boldsymbol{x}_{r_3}) \\ \qquad\qquad \text{if } o \in S \\ \boldsymbol{x}_o + F_o * (\boldsymbol{x}_{\text{better}} - \boldsymbol{x}_o) + F_o * (\boldsymbol{x}_{r_2} - \boldsymbol{x}_{r_3}) \\ \qquad\qquad \text{if } o \in I, \end{cases} \tag{10}$$

where o denotes the base vector, indices $r_1 \neq r_2 \neq r_3 \neq o$ are selected randomly from $[1,\ N]$, and *better* is a randomly selected index from the superior part of the population S. The base vector index is dependent on the current stage, i.e. $o = i$ in the first stage, and o is selected randomly in the second stage. The last individual in the mutation strategy $\boldsymbol{x}_{r_3}$ is perturbed with small probability p_d to help the base vector move out of the local area. This perturbation is performed by:

$$x_{r_3,j} = \begin{cases} a_j + rand(0,1) \times (b_j - a_j), \\ \qquad\qquad \text{if } rand_j(0,1) < p_d \\ x_{r_3,j} \quad \text{otherwise,} \end{cases} \tag{11}$$

where $p_d = 0.1 \times ps$ and a_j, b_j are the lower and the upper boundaries of the jth coordinate. After mutation, a binomial crossover produces new solutions. The parent solution is replaced if a new solution has a lower or equal function value.

C. CMA-ES

The only EA which is not based on DE principles is called Evolutionary Strategy with Covariance Matrix Adaptation (CMA-ES) and it was proposed by Hansen and Ostermeier [17]. A new point $\boldsymbol{x}^N$ is generated using a mutation strategy by adding the random vector to the current point $\boldsymbol{x}^E$:

$$\boldsymbol{x}^N = \boldsymbol{x}^E + \sigma \boldsymbol{B} N(\boldsymbol{0}, \boldsymbol{I}). \tag{12}$$

where B is matrix of size $D \times D$, $\boldsymbol{B}N(\boldsymbol{0}, \boldsymbol{I})$ is a linear transformation of $N(\boldsymbol{0}, \boldsymbol{I})$. Choosing B in a suitable way, any normal distribution with a zero mean vector can be generated by this transformation. The parameter of σ controls overall variance of the base vector in the mutation.

D. jSO

In 2017, Brest et al. proposed a variant of jSO derived from the SHADE [18], L-SHADE [19], and iL-SHADE [20] algorithms. In 2017, a variant of L-SHADE with the cooperation of the CMA-ES optimiser was proposed [21]. jSO uses a linear reduction of the population size with an initial size $N = 25 \times \sqrt{D} \times \log D$. The parameter p controlling part of a better solution for mutation is decreasing linearly from 0.25 to 0.125. The historical circle memories for the control parameters have to size $H = 5$. The initial mean values of the control parameters are $\mu_F = 0.3$, and $\mu_{\mathrm{CR}} = 0.8$ (the last positions are set to 0.9). jSO uses an enhanced current-to-$p$best mutation strategy is here controlled by a weighted F_w parameter:

$$\boldsymbol{u}_i = \boldsymbol{x}_i + F_w(\boldsymbol{x}_{\mathrm{pBest}} - \boldsymbol{x}_i) - F(\boldsymbol{x}_{\mathrm{r1}} - \boldsymbol{x}_{\mathrm{r2}}), \tag{13}$$

where

$$F_w = \begin{cases} 0.7F, & \textit{FES} < 0.2\textit{maxFES} \\ 0.8F, & \textit{FES} < 0.4\textit{maxFES} \\ 1.2F, & \text{otherwise}, \end{cases} \tag{14}$$

where FES is the current number of function evaluations, and *maxFES* is the maximal number of function evaluations. A binomial crossover is used to generate a new solutions. The values of F and *CR* are updated based on the successful previous settings and the archive of size $N \times 2.6$.

III. A Cooperative Model of Evolutionary Algorithms and Eigen Transformation

A cooperative model of four EAs (CoBiDE [13], IDEbd [16], CMA-ES [17], and jSO [19]) was introduced in 2019 [11]. Here, the original model is enhanced by the Eigen approach from CoBiDE used in all employed DE methods (a variant of CMA-ES uses a similar Eigen transformation). The idea of the proposed cooperative model is as follows.

One generation of a population P of individuals is developed by one of the employed EAs based on a roulette wheel with equal probabilities at the beginning. The method with the best results (based on new-good solutions) is more preferred in the next generation. This selection mechanism was inspired by the competitive DE variant [22].

Previous experiments show the linear population size reduction mechanism (jSO) performs well. Therefore, the population size of the proposed model is decreased linearly. The proposed cooperative model of Evolutionary Algorithms with a linear population size reduction and Eigen transformation is called *EA4eig*, and its pseudo-code is depicted in Algorithm 1.

Algorithm 1 Cooperative model of EAs: EA4eig

```
initialise population P = {x_1, x_2, ..., x_N}
evaluate all individuals by a goal function
set all probabilities equally q_h = 1/H
while stopping condition not reached do
  select a proper EA to generate a new generation
  for i = 1, 2, ..., N do
    create a new trial point y_i
    evaluate f(y_i)
    if current EA is CMA-ES then
      if f(y_i) ≤ f(x_worst) then
        insert y_i into Q
      else
        insert x_worst into Q
      end if
    else
      if f(y_i) ≤ f(x_i) then
        insert y_i into Q
      else
        insert x_i into Q
      end if
    end if
  end for
  P ← Q
  update population size N
  g = g + 1
end while
```

In the beginning, a population of N individuals is initialised in a search area, and it is evaluated by an objective function f. All the parameters of employed EAs in the model are initialised, and probabilities to use each hth algorithm are set equally to $q_h = 1/4$. Then, a proper EA to evaluate the population is selected randomly, using $q_h,\ h = 1, 2, \ldots, 4$. An algorithm with a higher probability has a higher chance to be used in the next generation. If the used algorithm produces successful individuals (better than the parents), the count of success is increased proportionally, and, the probabilities q_h are updated at the end of the generation:

$$q_h = \frac{n_h + n_0}{\sum_{j=1}^{H}(n_j + n_0)}, \tag{15}$$

where n_h is the count of the hth EA success, and $n_0 > 0$ is a constant to prevent a dramatic change of q_h by random

successful use of any employed EA. To avoid degeneration of the selection process, the values of q_h are initialised if any probability q_h is lower than the given limit $\delta > 0$. After each generation, the appropriate population size is computed, and if it is necessary, the population size N is reduced:

$$N = \mathit{round}[(\frac{N_{\min} - N_{\text{init}}}{\mathit{maxFES}})\mathit{FES} + N_{\text{init}}], \qquad (16)$$

where N_{init} is the initial value of population size, $N_{\min}$ is the final value of population size at the end of the search process, *FES* is the current objective function evaluations, and *maxFES* function evaluations allowed for the run.

CMA-ES has not stored and updated population os size N as DE variants. Here, only a weighted centre (seed) of the population is computed and used for the generation of CMA-ES. The individuals are generated from the same seed-position, perturbed by the mutation as defined with (12). The different population-approach of CMA-ES generates a special condition when the population of CMA-ES is relayed to DE variants.

IV. Experiments

The test suite of 12 problems was proposed for a special session and competition on Single Objective Bound Constrained Real-Parameter Numerical Optimisation, a part of Congress on Evolutionary Computation (CEC) 2022. This session was proposed for a competition of newly introduced optimisation algorithms. The test functions are described in [23], including the experimental settings required for the competition. The benchmark functions can be used at $D = 10,\ 20$.

All the control parameters of the EAs used in EA4eig are set to recommended values. Initial population size N_{init} is set to 100, minimum population size $N_{\min}$ is set to 10.

The *EA4eig* algorithm is implemented in Matlab 2020b, and this environment was also used for experiments. All computations were carried out on a standard PC with Windows 10, Intel(R) Core(TM)i7-9700 CPU 3.0 GHz, 16 GB RAM.

The performance of the proposed EA4eig is compared with several state-of-the-art variants of DE. At first, the original jSO is used in the comparison. In 2019, the variant of adaptive jDE100 using two independent populations was proposed for the competition CEC 2019 [24]. In 2020, the variant of adaptive j2020 was proposed for the CEC 2020 competition, which was derived from the jDE100 [25]. In 2021, the variant of adaptive j21 was proposed for the CEC 2021 competition, it extends the variant of j2020 [26]. Details of these algorithms are provided in the references, the algorithms used the recommended settings in the experiments.

V. Results

The basic characteristics from the results of the proposed cooperative model of Evolutionary Algorithms with linearly decreasing population size and Eigen transformation (EA4eig) are in Table I. It is obvious that the proposed EA4eig is able to solve twelve problems out of 24 (achieve minimal error value less than 1×10^{-8}). The worst results are provided for problems $F9$-$F12$, these problems are defined by composition functions. Only for problem $F11$ EA4eig provided sufficient accuracy for $D = 10$. For hybrid problems $F6$ and $F8$, EA4eig achieved only promising results, worse for the higher dimension. The time complexity of EA4eig is depicted in Table II, where the most important criterion is in the last column, for each dimension. The results of EA4eig are compared with four state-of-the-art DE variants using the Friedman test on medians of error values (accuracy) and achieved final values of *FES* (speed). The null hypothesis on equivalent efficiency of the algorithms was rejected in both aspects of the search with $p < 5 \times 10^{-6}$. This test provides mean ranks for all five methods computed from all 12 test problems, for each dimension independently. A lower mean rank means a better performing algorithm through all test problems (Table III). Columns marked '*avg*' present average mean ranks of each

TABLE I
Basic characteristics of EA4eig.

D	fun	min	max	med	mean	SD
10	1	5.63E-09	9.97E-09	8.65E-09	8.31E-09	1.34E-09
10	2	6.32E-09	3.98658	9.72E-09	1.46175	1.95395
10	3	5.99E-09	1.00E-08	8.70E-09	8.59E-09	9.98E-10
10	4	5.74E-09	3.97984	9.95E-01	1.26028	1.04298
10	5	4.42E-09	9.95E-09	8.01E-09	8.04E-09	1.62E-09
10	6	5.75E-04	1.48E-01	5.41E-03	1.74E-02	3.57E-02
10	7	5.78E-09	9.97E-09	8.72E-09	8.54E-09	1.17E-09
10	8	3.20E-04	2.64E-01	4.67E-02	7.09E-02	6.81E-02
10	9	185.502	185.502	185.502	185.502	5.78E-14
10	10	100.084	100.213	100.1585	100.1565	3.60E-02
10	11	5.55E-09	9.97E-09	9.51E-09	9.10E-09	1.06E-09
10	12	145.295	158.55	145.662	147.378	3.90145
20	1	6.39E-09	9.99E-09	9.29E-09	8.74E-09	1.14E-09
20	2	7.61E-09	3.98662	9.72E-09	1.06310	1.79308
20	3	5.79E-09	9.94E-09	9.44E-09	9.14E-09	9.38E-10
20	4	3.97984	19.8992	6.96471	8.68931	4.08091
20	5	6.02E-09	1.00E-08	9.34E-09	9.07E-09	8.89E-10
20	6	2.92E-02	4.42E-01	1.06E-01	1.49E-01	1.16E-01
20	7	8.41E-09	20	2.30209	3.50430	4.77194
20	8	2.91E-01	21.0441	20.2613	16.6196	7.47147
20	9	165.344	165.344	165.344	165.344	2.89E-14
20	10	100.201	223.252	100.256	108.259	30.4765
20	11	300	400	300	323.333	43.0183
20	12	188.675	200.005	200.004	199.626	2.06839

TABLE II
Time-complexity of proposed EA4eig.

D	T0	T1	T2	(T2-T1)/T0
10	0.0619	0.28	0.47	2.07
20	0.0625	0.34	1.27	14.88

TABLE III
Mean ranks from the Friedman tests from results of accuracy and speed.

	Accuracy (error)				Speed (FES)		
Alg.	D=10	D=20	avg	AVG	D=10	D=20	avg
EA4eig	**2.17**	**2**	**2.08**	2.33	2.5	2.67	2.58
j2020	3.46	3.63	3.54	2.81	**2.13**	**2.04**	**2.08**
j21	3.08	3.38	3.23	2.89	2.71	2.38	2.54
jSO	3.17	2.38	2.78	3.26	4.08	3.42	3.75
jDE100	3.13	3.63	3.38	3.71	3.58	4.5	4.04

method for accuracy or speed, regarding both dimensions. The column labelled '*AVG*' contains average mean ranks for each algorithm regarding accuracy and speed for both dimensions (the algorithms are ordered using values of *AVG*).

The proposed EA4eig achieved the best mean ranks in the case of accuracy for both dimensions, and it is outperformed by variant of j2020 (and j21 for $D = 20$) in speed.

TABLE IV
MEDIAN VALUES OF ALGORITHMS IN COMPARISON WITH SIGNIFICANCE FROM THE KRUSKAL-WALLIS TESTS, $D = 10$.

F	sig.	EA4eig	jDE100	j2020	j21	jSO
1	***	8.65E-09	**3.91E-09**	8.34E-09	8.73E-09	8.32E-09
2	***	**9.72E-09**	3.98658	3.98658	3.98658	3.98658
3	***	8.70E-09	**1.91E-09**	7.14E-09	5.25E-09	9.04E-09
4	***	**9.95E-01**	6.96471	5.96975	4.86155	2.98488
5	***	8.01E-09	**3.11E-09**	9.48E-09	9.59E-09	8.79E-09
6	***	**5.41E-03**	1.38750	3.51E-01	2.20E-01	2.77E-01
7	***	8.72E-09	**4.99E-09**	7.97E-09	8.53E-09	9.85E-09
8	***	**4.67E-02**	3.47E-01	1.50E-01	4.93E-02	1.82E-01
9	***	**185.502**	229.284	229.284	229.284	229.284
10	***	**100.159**	100.277	100.26	100.24	100.188
11	***	9.51E-09	**3.34E-09**	8.08E-09	7.11E-09	8.99E-09
12	***	**145.662**	163.516	163.506	162.7	162.7
$\sum$		6	5	0	0	0

More detail of algorithms' comparison provide median values and results of the Kruskal-Wallis tests in Table IV and V. Symbol of 'sig.' represents the level of significance (where '***' denotes $p < 0.001$). Proposed EA4eig achieved the best results especially for hybrid problems.

TABLE V
MEDIAN VALUES OF ALGORITHMS IN COMPARISON WITH SIGNIFICANCE FROM THE KRUSKAL-WALLIS TESTS, $D = 20$.

F	sig.	EA4eig	jDE100	j2020	j21	jSO
1	***	9.29E-09	**7.27E-09**	9.21E-09	9.29E-09	9.15E-09
2	***	**9.72E-09**	49.0845	49.0845	49.0845	44.8955
3	***	9.44E-09	**4.87E-09**	8.62E-09	5.32E-09	9.49E-09
4	***	**6.96471**	23.3815	16.5283	13.9294	**6.96471**
5	**	9.34E-09	8.97E-09	9.52E-09	9.64E-09	**8.89E-09**
6	***	**1.06E-01**	26.5025	7.47328	3.63594	4.96E-01
7	**	**2.30209**	4.57964	4.73498	3.42647	2.69638
8	***	**20.2613**	21.3664	21.2339	21.13	20.3003
9	***	**165.344**	180.781	180.781	180.781	180.781
10	***	100.256	100.3455	100.303	100.268	**100.234**
11	**	300	300	300	300	300
12	***	**200.004**	233.984	232.144	232.26	232.26
$\sum$		7	2	0	0	3

Table VI provides numbers of the best, second, third, and last positions of algorithms from the Kruskal-Wallis tests. Obviously, proposed EA4eig achieved mostly the first position, and it is the worst performing only in two problems.

Moreover, the Wilcoxon rank-sum test was applied to compare the proposed EA4eig with each of the state-of-the-art method on each problem and dimension (Table VII). EA4eig performs better ($-$) than jSO in 12 and worse ($+$) in three problems ($\approx$ is for similar results); it performs better than j21 in 15 and worse in three problems; it performs better than j2020 in 16, and worse in four problems; it performs better than jDE100 in 15 and worse in seven problems.

TABLE VI
NUMBER OF 1ST, 2ND, 3RD, AND LAST MEAN-RANK OF THE ALGORITHMS FROM THE KRUSKAL-WALLIS TESTS.

Position	Ea4eig	jDE100	jSO	j21	j2020
1st	12	6	4	2	0
2nd	3	2	10	6	1
3rd	4	1	2	7	8
last	2	12	3	3	2

TABLE VII
SIGNIFICANCE ACHIEVED FROM THE WILCOXON TESTS COMPARING EA4EIG WITH OTHER METHODS.

D	Fun	vs. jSO	vs. j21	vs. j2020	vs. jDE100
10	1	$\approx$	$\approx$	$\approx$	$+++$
10	2	$---$	$--$	$---$	$---$
10	3	$\approx$	$---$	$+++$	$+++$
10	4	$---$	$+++$	$---$	$---$
10	5	$\approx$	$--$	$--$	$+++$
10	6	$---$	$---$	$---$	$---$
10	7	$-$	$\approx$	$+$	$+++$
10	8	$---$	$\approx$	$-$	$---$
10	9	$---$	$---$	$---$	$---$
10	10	$--$	$---$	$---$	$---$
10	11	$\approx$	$+++$	$+++$	$+++$
10	12	$---$	$---$	$---$	$---$
b/s/w		0/4/8	2/3/7	3/1/8	5/0/7
20	1	$\approx$	$\approx$	$\approx$	$+++$
20	2	$---$	$---$	$---$	$---$
20	3	$\approx$	$+++$	$++$	$+++$
20	4	$\approx$	$---$	$---$	$---$
20	5	$+$	$\approx$	$\approx$	$\approx$
20	6	$---$	$---$	$---$	$---$
20	7	$\approx$	$-$	$-$	$---$
20	8	$\approx$	$---$	$---$	$---$
20	9	$---$	$---$	$---$	$---$
20	10	$+++$	$\approx$	$\approx$	$---$
20	11	$++$	$-$	$-$	$\approx$
20	12	$---$	$---$	$---$	$---$
b/s/w		3/5/4	1/3/8	1/3/8	2/2/8

Finally, usage of the employed EAs in the EA4eig provided Table VIII, where success (in %) is presented. Low success of CMA-ES is caused by different approach when CMA-ES manipulate with the population. Higher success of some method in some problems confirms the No-Free-Lunch theorem [1].

Different insight into behaviour of the algorithms in the comparison provide convergence plots depicted in Figure 1 and 2. The compared methods converge similarly in most test problems, in some problems ($F9$, $F10$, $F12$) the EA4eig converges with a lower error or in the earlier stage.

VI. CONCLUSION

The proposed model of EA4eig was proposed and experimentally compared with four state-of-the-art DE variants on the CEC 2022 benchmark set. Achieved results illustrate the high performance of the proposed method when achieving the best results in 12 out of 24 test problems. Performance of EA4eig is higher, especially in the case of hybrid problems, which represent real-valued problems. Tuning of settings and

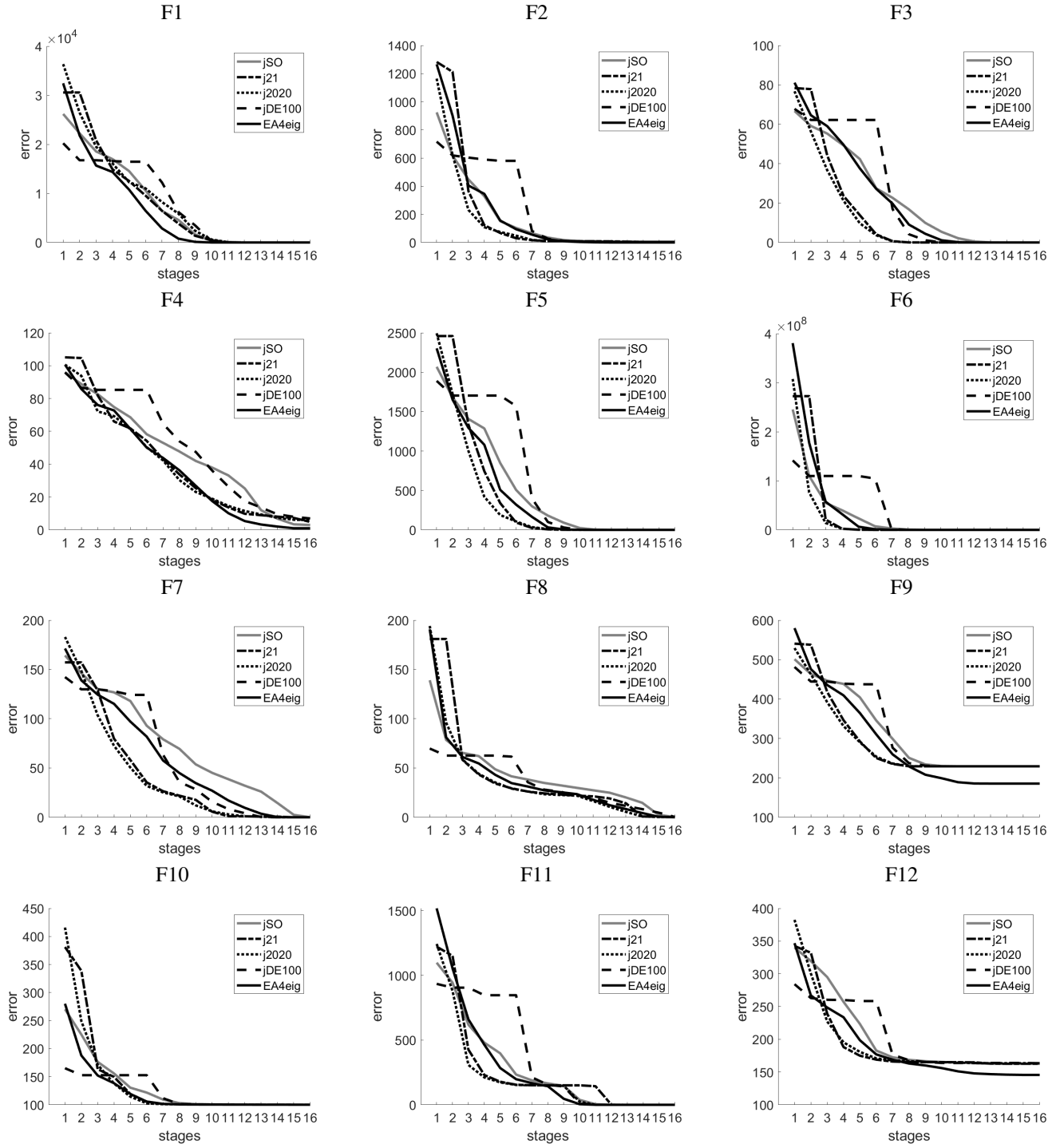

Fig. 1. Convergence error-lines of compared algorithms, $D = 10$.

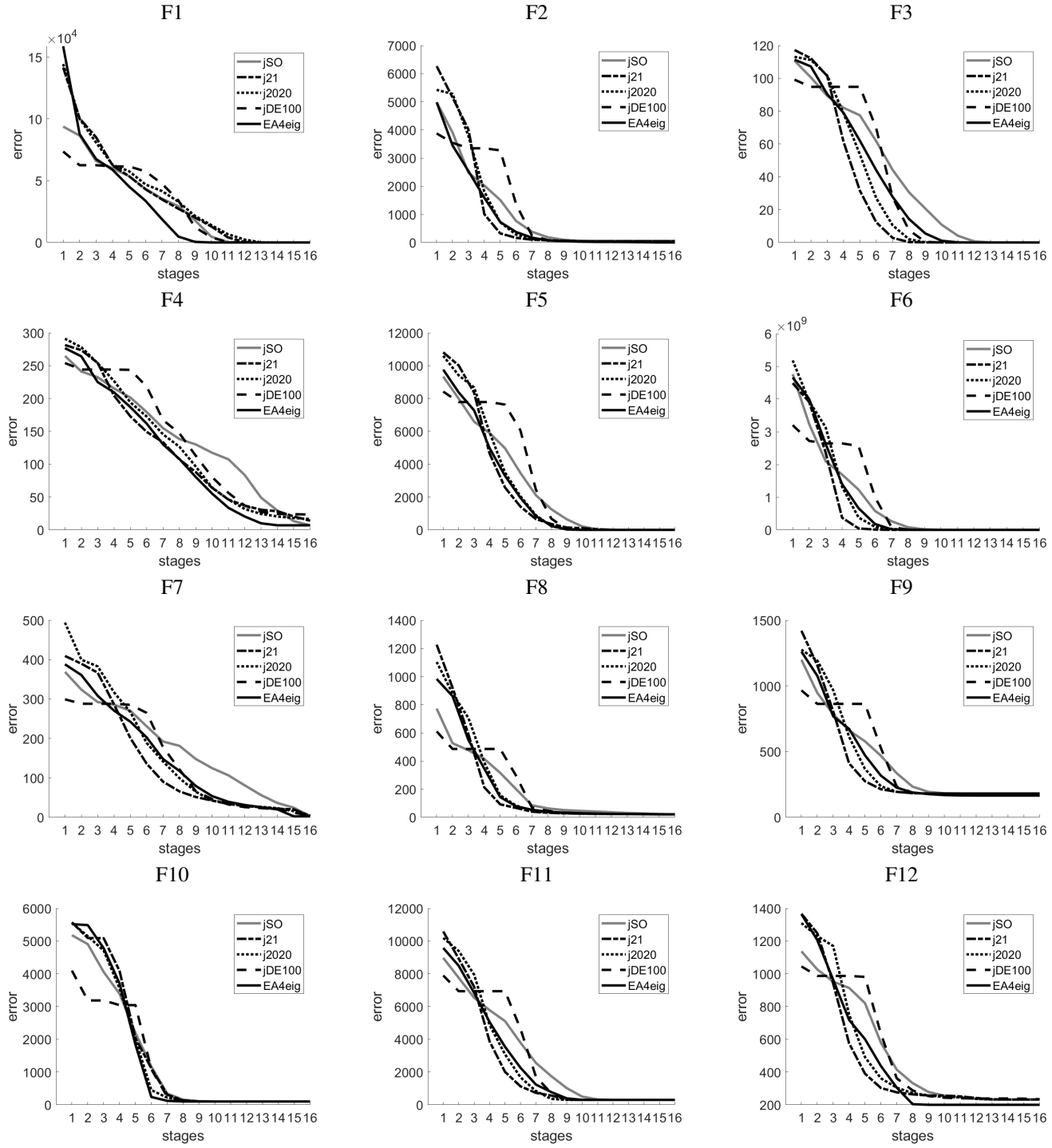

Fig. 2. Convergence error-lines of compared algorithms, $D = 20$.

TABLE VIII
PERCENTAGE SUCCESS RATE OF EMPLOYED EAS IN PROPOSED EA4EIG.

D	Fun	CoBiDE	IDE	CMAES	jSO
10	1	18	**60**	0	21
10	2	17	**56**	0	28
10	3	32	**57**	1	11
10	4	**48**	47	1	4
10	5	19	**51**	1	28
10	6	30	**39**	1	30
10	7	**53**	38	2	7
10	8	**39**	34	1	26
10	9	4	9	3	**84**
10	10	16	25	18	**40**
10	11	23	**51**	0	26
10	12	3	7	3	**88**
Avg.		25.2	39.5	2.6	32.8
20	1	14	**49**	1	37
20	2	6	15	0	**79**
20	3	28	**60**	0	12
20	4	**51**	47	0	1
20	5	15	**56**	8	21
20	6	26	34	2	**38**
20	7	**63**	26	1	11
20	8	29	19	0	**51**
20	9	4	6	1	**89**
20	10	33	**34**	3	30
20	11	47	**50**	0	3
20	12	3	6	0	**91**
Avg.		26.6	33.5	1.3	38.6

combination of employed EAs in EA4eig will be studied in future research.

ACKNOWLEDGMENT

The research was supported by the University of Ostrava from the project SGS17/PřF-MF/2022.

REFERENCES

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

[2] M. A. Potter and K. A. D. Jong, "A cooperative coevolutionary approach to function optimization," in *Proceedings of the Third Conference on Parallel Problem Solving from Nature (PPSN 94)*, 1994.

[3] X. Ma, X. Li, Q. Zhang, K. Tang, Z. Liang, W. Xie, and Z. Zhu, "A survey on cooperative co-evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 421–441, 2019.

[4] H. Cai, B. Liu, and S. Pan, "On the cooperation between evolutionary algorithms and constraint handling techniques: A further empirical study," *IEEE Access*, vol. 8, pp. 130 598–130 606, 2020.

[5] J. G. Falcón-Cardona, H. Ishibuchi, C. A. Coello Coello, and M. Emmerich, "On the effect of the cooperation of indicator-based multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 4, pp. 681–695, 2021.

[6] M. M. Fouad, A. I. El-Desouky, R. Al-Hajj, and E.-S. M. El-Kenawy, "Dynamic group-based cooperative optimization algorithm," *IEEE Access*, vol. 8, pp. 148 378–148 403, 2020.

[7] P. Bujok and J. Tvrdík, "Parallel migration model employing various adaptive variants of differential evolution," in *Swarm and Evolutionary Computation*, ser. Lecture Notes in Computer Science, L. Rutkowski, M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. A. Zadeh, and J. M. Zurada, Eds., vol. 7269. Springer Berlin Heidelberg, 2012, pp. 39–47.

[8] P. Bujok, "Migration model of adaptive differential evolution applied to real-world problems," in *Artificial Intelligence and Soft Computing - Part I*, ser. Lecture Notes in Computer Science, vol. 10841, 2018, pp. 313–322, 17th International Conference on Artificial Intelligence and Soft Computing ICAISC, Zakopane, Poland, 2018.

[9] R. Poláková, J. Tvrdík, and P. Bujok, "Cooperation of optimization algorithms: A simple hierarchical model," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, 2015, pp. 1046–1052.

[10] P. Bujok, "Cooperative model for nature-inspired algorithms in solving real-world optimization problems," in *Bioinspired Optimization Methods and Their Applications*, P. Korošec, N. Melab, and E.-G. Talbi, Eds. Cham: Springer International Publishing, 2018, pp. 50–61.

[11] P. Bujok and A. Zamuda, "Cooperative model of evolutionary algorithms applied to CEC 2019 single objective numerical optimization," in *2019 IEEE Congress on Evolutionary Computation (CEC)*, 2019, pp. 366–371.

[12] P. Bujok, "Cooperative model of evolutionary algorithms and real-world problems," in *Swarm, Evolutionary, and Memetic Computing and Fuzzy and Neural Computing*, A. Zamuda, S. Das, P. N. Suganthan, and B. K. Panigrahi, Eds. Cham: Springer International Publishing, 2020, pp. 1–12.

[13] Y. Wang, H.-X. Li, T. Huang, and L. Li, "Differential evolution based on covariance matrix learning and bimodal distribution parameter setting," *Applied Soft Computing*, vol. 18, pp. 232–247, 2014.

[14] S.-M. Guo, J. S.-H. Tsai, C.-C. Yang, and P.-H. Hsu, "A self-optimization approach for L-SHADE incorporated with eigenvector-based crossover and successful-parent-selecting framework on CEC 2015 benchmark set," in *IEEE Congress on Evolutionary Computation (CEC)*, 2015, pp. 1003–1010.

[15] L. Tang, Y. Dong, and J. Liu, "Differential evolution with an individual-dependent mechanism," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 4, pp. 560–574, 2015.

[16] P. Bujok and J. Tvrdík, "Enhanced individual-dependent differential evolution with population size adaptation," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, June 2017, pp. 1358–1365.

[17] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation," in *Proceedings of IEEE International Conference on Evolutionary Computation*, May 1996, pp. 312–317.

[18] R. Tanabe and A. S. Fukunaga, "Evaluating the performance of SHADE on CEC 2013 benchmark problems," in *IEEE Congress on Evolutionary Computation 2013*. IEEE Computational Intelligence Society, 2013, pp. 1952–1959.

[19] J. Brest, M. S. Maučec, and B. Bošković, "Single objective real-parameter optimization: Algorithm jSO," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 1311–1318.

[20] J. Brest, M. S. Maučec, and B. Bošković, "iL-SHADE: Improved l-shade algorithm for single objective real-parameter optimization," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 1188–1195.

[21] A. W. Mohamed, A. A. Hadi, A. M. Fattouh, and K. M. Jambi, "Lshade with semi-parameter adaptation hybrid with cma-es for solving cec 2017 benchmark problems," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, 2017, pp. 145–152.

[22] J. Tvrdík, "Competitive differential evolution," in *MENDEL 2006, 12th International Conference on Soft Computing*, R. Matoušek and P. Ošmera, Eds. Brno: University of Technology, 2006, pp. 7–12.

[23] A. Kumar, K. V. Price, A. W. Mohamed, A. A. Hadi, and P. N. Suganthan, "Problem definitions and evaluation criteria for the cec 2022 special session and competition on single objective bound constrained numerical optimization," Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China And Technical Report, Nanyang Technological University, Singapore, Tech. Rep., 2021, github.com/P-N-Suganthan.

[24] J. Brest, M. S. Maučec, and B. Bošković, "The 100-digit challenge: Algorithm jDE100," in *2019 IEEE Congress on Evolutionary Computation (CEC)*, 2019, pp. 19–26.

[25] J. Brest, M. S. Maučec, and B. Bošković, "Differential evolution algorithm for single objective bound-constrained optimization: Algorithm j2020," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, 2020, pp. 1–8.

[26] J. Brest, M. S. Maucec, and B. Boskovic, "Self-adaptive differential evolution algorithm with population size reduction for single objective bound-constrained optimization: Algorithm j21," in *IEEE Congress on Evolutionary Computation, CEC 2021, Kraków, Poland, June 28 - July 1, 2021*. IEEE, 2021, pp. 817–824.