Evidence of correspondence

Evolutionary Rao Algorithm

1. **First submission (04 January 2021)**
2. **LoA with Major Revision (23 February 2021)**
3. **Responses to Reviewers, Final submission (12 March 2021)**
4. **LoA with Fully Accepted (04 April 2021)**
5. **Proof Reading (11 April 2021)**

Telkom University

SUYANTO SUYANTO <suyanto@telkomuniversity.ac.id>

Confirming submission to Journal of Computational Science

1 message

Journal of Computational Science <em@editorialmanager.com> Mon, Jan 4, 2021 at 2:38 PM
Reply-To: Journal of Computational Science <jocs@elsevier.com>
To: Suyanto Suyanto <suyanto@telkomuniversity.ac.id>

This is an automated message.

Evolutionary Rao Algorithm

Dear Dr. Suyanto,

We have received the above referenced manuscript you submitted to Journal of Computational Science.

To track the status of your manuscript, please log in as an author at https://www.editorialmanager.com/jocsci/, and navigate to the "Submissions Being Processed" folder.

Thank you for submitting your work to this journal.

Kind regards,
Journal of Computational Science

More information and support

You will find information relevant for you as an author on Elsevier's Author Hub: https://www.elsevier.com/authors

FAQ: How can I reset a forgotten password?
https://service.elsevier.com/app/answers/detail/a_id/28452/supporthub/publishing/
For further assistance, please visit our customer service site: https://service.elsevier.com/app/home/supporthub/publishing/
Here you can search for solutions on a range of topics, find answers to frequently asked questions, and learn more about Editorial Manager via interactive tutorials. You can also talk 24/7 to our customer support team by phone and 24/7 by live chat and email

Journal of Computational Science

Evolutionary Rao Algorithm

--Manuscript Draft--

Manuscript Number:	
Article Type:	Full Length Article
Keywords:	evolutionary Rao algorithm; evolutionary operators; exploitative; explorative; two populations
Corresponding Author:	Suyanto Suyanto Telkom University Bandung, INDONESIA
First Author:	Suyanto Suyanto
Order of Authors:	Suyanto Suyanto
	Agung Toto Wibowo
	Said Al Faraby
	Siti Sa'adah
	Rita Rismala
Abstract:	In this paper, an evolutionary Rao algorithm (ERA) is proposed to enhance three state-of-the-art metaheuristic Rao algorithms (Rao-1, Rao-2, Rao-3), by introducing two new schemes. Firstly, the population is split into two sub-populations based on their qualities: high and low, with a particular portion that can be simply tuned depending on the given problem. The high-quality sub-population searches for an optimum solution in an exploitative manner using a movement scheme used in the Rao-3 algorithm. Meanwhile, the low-quality one does in an explorative fashion using a new random walk. Secondly, two evolutionary operators: crossover and mutation, are exploited to make the proposed ERA faster in the exploitative and explorative searching, respectively. Here, both operators are implemented using a random scheme with the common probabilistic values so that they do not create any additional parameters. Examination of the twenty-three benchmark functions: seven unimodal, six multimodal, and ten fixed-dimension multimodal shows that the proposed ERA outperforms the three original Rao algorithms. A detailed investigation indicates that both introduced schemes work very well to make the ERA evolves faster in an exploitative manner, which is created by a high portion of high-quality individuals and the crossover operator, and avoids trapping on the local optimum solutions in an explorative manner, which is created by a high portion of low-quality individuals and the mutation operator.
Suggested Reviewers:	Tiebin Wu wutiebin81@csu.edu.cn Tiebin Wu is doing some researches about metaheuristic algorithms.
	Hu Peng hu_peng@whu.edu.cn Hu Peng is interested in the swarm intelligences area.
Opposed Reviewers:	

January 04, 2021

Dear Peter Sloot,

I wish to submit a full manuscript entitled “Evolutionary Rao Algorithm” for consideration by the Journal of Computational Science. The manuscript has been checked using both Grammarly Premium and iThenticate with a low similarity index of 19% without exclude any source.

This manuscript is written based on our original research funded by the Directorate of Research and Community Service PPM, Telkom University, with grant number: 444/PNLT3/PPM/2020. In this manuscript, we propose an evolutionary Rao algorithm (ERA) to enhance three state-of-the-art metaheuristic Rao algorithms (Rao-1, Rao-2, Rao-3). Two new schemes are introduced. Firstly, the population is split into two sub-populations based on their qualities: high and low, with a particular portion that can be simply tuned depending on the given problem. Secondly, two evolutionary operators: crossover and mutation, are exploited to make the proposed ERA faster in the exploitative and explorative searching, respectively. Examination of the twenty-three benchmark functions: seven unimodal, six multimodal, and ten fixed-dimension multimodal shows that the proposed ERA outperforms the three original Rao algorithms. A detailed investigation indicates that both introduced schemes work very well to make the ERA evolves faster, in an exploitative manner that is created by a high portion of high-quality individuals and the crossover operator and avoids trapping on the local optimum solutions, in an explorative manner that is created by a high portion of low-quality individuals and the mutation operator.

Thank you for your consideration of this manuscript. Please address all correspondence concerning this manuscript to me at suyanto@telkomuniversity.ac.id.

Sincerely,

Suyanto
Telkom University
Jl. Telekomunikasi Terusan Buah Batu Bandung 40257, Indonesia

Highlights

In this manuscript, an evolutionary Rao algorithm (ERA) is proposed to enhance three state-of-the-art metaheuristic Rao algorithms by introducing two new schemes. The first scheme: splitting the population into two sub-populations of high and low quality individuals with a proper portion for the given problem, increases the evolution speed and the accuracy. The Second scheme: incorporating both crossover and mutation operators makes ERA faster in the exploitative and explorative searching, respectively. Examination of the twenty-three benchmark functions: seven unimodal, six multimodal, and ten fixed-dimension multimodal shows that the proposed ERA outperforms the three original Rao algorithms. A detailed investigation indicates that both schemes work very well as they designed.

Evolutionary Rao Algorithm

Abstract

In this paper, an evolutionary Rao algorithm (ERA) is proposed to enhance three state-of-the-art metaheuristic Rao algorithms (Rao-1, Rao-2, Rao-3), by introducing two new schemes. Firstly, the population is split into two sub-populations based on their qualities: high and low, with a particular portion that can be simply tuned depending on the given problem. The high-quality sub-population searches for an optimum solution in an exploitative manner using a movement scheme used in the Rao-3 algorithm. Meanwhile, the low-quality one does in an explorative fashion using a new random walk. Secondly, two evolutionary operators: crossover and mutation, are exploited to make the proposed ERA faster in the exploitative and explorative searching, respectively. Here, both operators are implemented using a random scheme with the common probabilistic values so that they do not create any additional parameters. Examination of the twenty-three benchmark functions: seven unimodal, six multimodal, and ten fixed-dimension multimodal shows that the proposed ERA outperforms the three original Rao algorithms. A detailed investigation indicates that both introduced schemes work very well to make the ERA evolves faster in an exploitative manner, which is created by a high portion of high-quality individuals and the crossover operator, and avoids trapping on the local optimum solutions in an explorative manner, which is created by a high portion of low-quality individuals and the mutation operator.

Keywords: evolutionary Rao algorithm, evolutionary operators, exploitative, explorative, two populations

1. Introduction

The metaheuristic optimization algorithms that can be categorized into two groups: evolutionary algorithms (EAs) and swarm intelligence (SI) algorithms [1]. EAs are inspired by both evolution and natural selection, such as Genetic Algorithm (GA) [2], [3], Evolution Strategies (ES) [4], [5], and Differential Evolution (DE) [6]. Meanwhile, SI algorithms are inspired by a natural swarm, such as Particle Swarm Optimization (PSO) [7], [8], Firefly Algorithm (FA) [9], [10], Grey Wolf Optimizer (GWO) [11], [12], and Ant Lion Optimization (ALO) [13].

GA is one of the most popular EAs introduced in the 1970s [14]. It uses evolution and natural selection applied to its population over generations. A population consists of some individual chromosomes, each representing a candidate solution. The new chromosomes in a generation are either some of the best chromosomes (elitism) in the previous generation or are generated by some genetic operations, such as crossover and mutation. The crossover takes two chromosomes and produces one offspring inherited part of chromosome values from each of the parents. In contrast, the mutation is randomly changing some values in a chromosome. The crossover and mutation are responsible for exploration, while elitism directs toward exploitation. GA has an ability to avoid being trapped in the local optima. It is also applicable to non-differentiable and high dimensionality functions. On the other hand, it converges slowly because of the highly-random operations that do not give a clear direction to find the global optimum solution quickly. However, various improvement schemes have been proposed to overcome the drawback, such as a concept of human-like constrained-mating [15] that creates a more explorative search strategy.

In 1995, the Particle Swarm Optimisation (PSO) was introduced by Kennedy and Elberhart [16]. The movements of the particles in searching for a global optimum mimics the behavior of bird flocking and fish schooling. PSO is one of the most popular SI algorithms since it has three advantages: easy to implement, few parameters that are simply tuned, and effective in searching the global optimum solution since it has a clearer direction than GA. However, it tends to prematurely converge on a local optimum in optimizing a multimodal function since it uses a static finite leader and group based on a linear movement. Therefore, some strategies are developed to tackle the issue, such as a learning structure [17] to decouple exploration and exploitation and a dynamic updating of the inertia weights [18] to control the convergence.

In 2009, the Firefly Algorithm (FA) was proposed [19]. In FA, each firefly will be attracted to all other brighter (better) fireflies, not only to the global best like in PSO. In addition, the attractiveness of a brighter firefly is decreased proportioned to the distance between the two fireflies due to the light absorption. Since the fireflies will usually be attracted more to their brighter neighbor than the further away brightest individual, the exploration is more effective than PSO. In other words, FA uses a dynamic leader and group based on a nonlinear movement. Moreover, FA can be turned into PSO by setting the light absorption parameter such that every firefly can be seen clearly by all other fireflies. Consequently, all fireflies will be attracted to the brightest one (global best). In some experiments, FA shows better performance than PSO due to two critical characteristics [20]: 1) FA usually divide its population into a subgroup, 2) By not having an explicit global best, FA can avoid premature convergence. To enhance the performance of FA, several improved schemes are created, such as a courtship learning framework [21], where the population is divided into sub-populations: female and male, to improve the convergence speed and solution accuracy; and a best neighbor guided strategy [22], where each firefly is attracted to the best firefly among some randomly chosen neighbors to decrease the firefly oscillations in every attraction-induced migration stage as well as increase the probability of the guidance a new better direction.

In 2014, Grey Wolf Optimization (GWO) was introduced by Mirjalili [23]. It is inspired by both the social hierarchy and hunting methods of grey wolves (GWs). The hierarchy of GWs has four groups: alpha, beta, delta, and omegas. GWO selects the three fittest wolves (best solutions) as the alpha, beta, and delta while the rest as omegas. The hunting of GWs is guided by the three fittest wolves. All omegas follow them. It has four phases, which are mathematically modeled into four behaviors: Harassing Prey, Hunting, Attacking, and Searching. They make GWO has a high exploitative searching strategy. It quickly converges to an optimum solution for unimodal functions. However, it suffers from multimodal functions since it has a low explorative movement. Therefore, some variants of GWO are developed by incorporating a differential evolution and elimination mechanism [24], combining a simulated annealing [25], adding a refraction learning operator [26], or introducing a dimension learning-based hunting movement strategy [27] that uses a different approach to construct a neighborhood for each wolf to enhances the balance between local and global search and maintains diversity.

In 2015, Ant Lion Optimizer (ALO) was proposed by Mirjalili [28]. ALO mimics the interaction between antlions and ants in the trap, where ants move over the search space and antlions hunt them and become fitter using traps. A new random walk is introduced to model the ant's movement as they move stochastically in nature to find some food. It has high exploitation and convergence speed because of both adaptive boundary shrinking mechanism and elitism. It also high exploration due to the random walk and roulette wheel selection mechanisms. However, although it has few parameters, some schemes and movements make ALO seems too-complicated. Hence, some versions of ALO are created by modifying, hybridizing, and providing an ability so solve a multi-objective problem [13].

In 2020, the metaphor-less optimization methods called Rao algorithms were proposed by Ravipudi Venkata Rao [29]. The Rao Algorithms use both best and worst solutions in each iteration, as well as the random interactions among the candidate solutions, to quickly find an optimum solution. They need two standard parameters: population size and a maximum number of evaluations, which are easy to adjust. They drop many parameters used in the previous metaphor-based algorithms, such as cohesion, intensity, probability, and other commonly challenging parameters to tune carefully.

The Rao algorithms have three variants: Rao-1, Rao-2, and Rao-3, which respectively use three different equations as follow

$$X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,best,i} - X_{j,worst,i}) \tag{1}$$

$$\begin{aligned} X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,best,i} - X_{j,worst,i}) + \\ r_{2,j,i}(|X_{j,k,i} \text{ or } X_{j,l,i}| - |X_{j,l,i} \text{ or } X_{j,k,i}|), \end{aligned} \tag{2}$$

$$\begin{aligned} X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,best,i} - |X_{j,worst,i}|) + \\ r_{2,j,i}(|X_{j,k,i} \text{ or } X_{j,l,i}| - (X_{j,l,i} \text{ or } X_{j,k,i})), \end{aligned} \tag{3}$$

where $X_{j,best,i}$ represents the best candidate as value of variable j, and $X_{j,worst,i}$ represents the worst candidate as value of variable j, both throughout the i-th iteration. $X'_{j,k,i}$ is the updated value after the equation, and both $r_{1,j,i}$ as well as $r_{2,j,i}$ are randomly generated in [0,1] for the j-th variable throughout the i-th iteration. In the term $|X_{j,k,i} \text{ or } X_{j,l,i}|$, the candidate solution k is compared to another candidate l, which is randomly selected from the available candidates in the population. The term $|X_{j,k,i}|$ is selected if k is fitter than l. Otherwise, the $|X_{j,l,i}|$ is chosen. The same rule is applied to the second the term $(X_{j,l,i} \text{ or } X_{j,k,i})$.

All formulas used in the three Rao algorithms are similar to GWO, which makes them more exploitative than explorative. Using both best and worst solutions, they converge to an optimum solution for unimodal functions more quickly than GWO. However, with low explorative movement, they can be worse for multimodal functions. As described in [29], Rao is easy to get stuck in multimodal functions. Rao-3 gives a better solution only in the Schwefel function from the six benchmark multimodal-functions and much worse for the other five benchmark multimodal-functions.

Therefore, in this research, an evolutionary Rao algorithm (ERA) is proposed to enhance the three original Rao algorithms by introducing two additional schemes. Firstly, the population is split into two sub-populations based on their qualities: high and low, with a particular portion depending on the given problem. The high-quality sub-population searches for an optimum solution in an exploitative manner using a movement scheme used in the Rao-3 algorithm. Meanwhile, the low-quality one does in an explorative fashion using a new random walk introduced in this research. This scheme is similar to the courtship learning framework in the Enhanced FA [21], where the population is also divided into two sub-populations: female and male, but ERA uses a pre-defined specific portion. Secondly, two evolutionary operators: crossover and mutation, are exploited to make the proposed ERA faster in the exploitative and explorative searching, respectively. Here, both operators are implemented using a random scheme with the common probabilistic values so that they do not create any additional parameters. The ERA is then examined using twenty-three benchmark functions: seven unimodal, six multimodal, and ten fixed-dimension multimodal, and compared to the three original Rao algorithms.

2. Proposed Evolutionary Rao Algorithm

The pseudo-code of ERA is illustrated in Algorithm 1. In the initial phase, define the population size p and the portion s, and initialize the population of p individuals. Next, in the second phase, an evolution is performed until a stopping condition is reached. In each generation, five steps are carried out. Firstly, the quality of each individual is calculated and their quality-ranks are then sorted in the descending mode. Secondly, the population is split into two sub-populations: high-quality (HQ) and low-quality (LQ), with the defined portion s, and both the best individual X_{best} and the worst individual X_{worst} are selected. Thirdly, each HQ individual is moved to follow the X_{best} using Eq. (3). Fourthly, the fittest HQ individual is selected as the BestHF, and then one of the two evolutionary operators is chosen: crossover (exploitative) or mutation (explorative), to move the X_{best}. Finally, each LQ individual is moved using a new random walk.

Algorithm 1: Evolutionary Rao Algorithm

Result: X_{best} as the optimum solution

Set p as the number of individuals (population size);

Set s as the portion of high-quality (HQ) individuals;

Initialization of p individuals;

while *StoppingCondition = false* **do**

  for each individual, calculate its quality and then sort the quality-ranks in the descending mode;

  Select the fittest individual as the X_{best};

  Select the most fit individuals with the defined portion s as the HQ and the rests as the low-quality (LQ) individuals;

  Select the lowest-quality individual as the X_{worst};

  for each HQ individual, move it to follow the X_{best} using Eq. (3);

  Select the fittest HQ individual as the BestHF;

  if *rand* $>$ *0.5* **then**

    Offsprings = Crossover(BestHF, X_{best});

    Replacement(BestHF, X_{best}, Offsprings);

  else

    Offspring = Mutation(X_{best});

    Replacement(X_{best}, Offspring);

  end

  for each LQ individual move it to follow or distract a randomly selected HQ individual based on Eq. (8);

end

2.1. Two sub-populations

The population of p candidate solutions (individuals) is split into two sub-populations based on their qualities: high and low, with a proper portion based on the given problem. The high-quality (HQ) sub-population searches for an optimum solution in an exploitative manner using the same movement scheme as in the Rao-3 algorithm. Meanwhile, the low-quality (LQ) one does in an explorative fashion using a new random walk introduced in this research. Hence, this scheme creates a new parameter s: the portion of the high and the low-quality individuals in the population, which is in the interval (0, 1) and easy to adjust. Hypothetically, it should be high (more than 0.5) to make ERA more exploitative and faster to solve the unimodal functions. In contrast, it must be low (less than 0.5) to make ERA more explorative to solve the multimodal functions.

The population of p individuals is split into the high-quality sub-population of h individuals and the low-quality sub-population of l individuals, which are calculated as

$$h = \lfloor (p-1) \times s \rfloor, \tag{4}$$

$$l = (p-1) - h, \tag{5}$$

where s is the portion of HQ individuals in the population.

However, both Eq. (4) and Eq. (5) may produce zero for either h or l if the portion s is too-small or too-high. Hence, an enforcement procedure is implemented to ensure that a too-small s makes the HQ sub-population consists of at least two individuals and a too-big s also makes the LQ sub-population contains at least two individuals.

2.2. Crossover

The crossover is implemented using a whole arithmetic crossover, which is defined as

$$\begin{aligned} X' &= \alpha \cdot X + (1-\alpha) \cdot Y \\ Y' &= \alpha \cdot Y + (1-\alpha) \cdot X \end{aligned} \tag{6}$$

where α is a constant in the interval (0, 1), which is randomly generated to be not equal to 0.5 to prevent generating the same two offsprings (new individuals);

if $\alpha = 0.5$, then both offsprings X' and Y' are the same as the average of both current individuals X and Y.

2.3. Mutation

The mutation is simply implemented using a creep mutation by adding a small value (positive or negative) to each mutated element. The small value is randomly generated using a Gaussian probability that is symmetric, distributed on 0, and has a high probability for the smaller values. The creep mutation is defined as

$$\langle x_1, x_2, ...x_n \rangle \rightarrow \langle x'_1, x'_2, ...x'_n \rangle, \tag{7}$$

where $x_1, x_2, ...x_n \in [L_i, U_i]$, L_i and U_i are the lower and upper bounds of the interval of the ith element.

2.4. Random walk

To provide an ability to search for an optimum solution in an explorative manner, each LQ individual is moved using a new random walk formulated as

$$X'_{m,LQ,i} = X_{m,LQ,i} + r_{1,m,i}(X_{m,HQ,n} - X_{m,LQ,i}) \tag{8}$$

where $X_{m,LQ,i}$ and $X_{m,HQ,n}$ is the LQ individual i and the HQ individual n (randomly selected from the high-quality sub-population), respectively, and m is the randomly selected dimension; not all dimensions is used here to make this random walk more explorative.

3. Results and Discussion

In this research, twenty-three benchmark functions: seven unimodal, six multimodal, and ten fixed-dimension multimodal functions as described in [29], are used to investigate both exploitation and exploration abilities of the proposed ERA. Table 1 illustrates the benchmark functions with their identities

(ID), names, types, dimensions, ranges, and global optimum values f_{min}. Seven benchmark functions, with ID = 1 to 7, are unimodal to examine the exploitation ability. Next, six benchmark functions, ID = 8 to 13, are multimodal, with many local optima that increase as the dimension increases, to evaluate the exploration ability. Finally, ten functions, ID = 14 to 23, are fixed-dimension multimodal to investigate the exploration ability in the case of fixed-dimension optimization problems.

3.1. Parameter tuning

Here, both parameters of the proposed ERA: population p and portion s, are independently tuned for the twenty three benchmark functions. For each function, ninety experiments are performed using combination of ten values of p = 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 and nine values of s = 0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 0.8, 0.9, which can be defined as pairs of (10, 0.1), (10, 0.2), ..., (100, 0.9). For each experiment, the maximum number of function evaluations is set to 30,000 with 10 runs to reduce the coincidence. Here, only three experimental results of the representative benchmark functions are shown and discussed: unimodal (Sphere, ID = 1), multimodal (Schwefel, ID = 8), and fixed-dimension multimodal (Shekel 7, ID = 22) since the results of 20 other benchmark functions are similar to those three results.

Figure 1 illustrates the experimental results for the problem of searching a minimum solution to a unimodal function of Sphere (ID = 1), where the vertical axis uses log(mean solution) to ensure the bar chart clearly shows all results from the ninety experiments. It can be seen that a too-small (10) or a big population p (30 to 100) makes the ERA produces a bad solution. The bigger the p the worse the solution. A small portion s (0.5 or less) also yields a poor solution. The smaller the s the worse the solution. Hence, the combination of a too-big p and a too-small s is not recommended. The optimum combination is reached on p = 20 and s = 0.8. This result proves that a big portion of high-quality individuals in the small population makes the proposed ERA more exploitative and faster to find the optimum solution.

Table 1: Twenty three benchmark functions: seven unimodal, six multimodal, and ten fixed-dimension multimodal (FDM) functions

ID	Function Name	Type	Dimension	Range	f_{min}
1	Sphere	Unimodal	30	[-100, 100]	0
2	Schwefel 2.22	Unimodal	30	[-100, 100]	0
3	Schwefel 1.2	Unimodal	30	[-100, 100]	0
4	Schwefel 2.21	Unimodal	30	[-100, 100]	0
5	Rosenbrock	Unimodal	30	[-30, 30]	0
6	Step	Unimodal	30	[-100, 100]	0
7	Quartic	Unimodal	30	[-1.28, 1.28]	0
8	Schwefel	Multimodal	30	[-500, 500]	0
9	Rastrigin	Multimodal	30	[-5.12, 5.12]	0
10	Ackley	Multimodal	30	[-32, 32]	0
11	Griewank	Multimodal	30	[-600, 600]	0
12	Penalized	Multimodal	30	[-50, 50]	0
13	Penalized2	Multimodal	30	[-50, 50]	0
14	Foxholes	FDM	2	[-65, 65]	0.998
15	Kowalik	FDM	4	[-5, 5]	0.0003
16	Six Hump Camel	FDM	2	[-5, 5]	-1.0316
17	Branin	FDM	2	[-5, 5]	0.398
18	GoldStein-Price	FDM	2	[-2, 2]	3
19	Hartman 3	FDM	3	[0, 1]	-3.86
20	Hartman 6	FDM	6	[0, 1]	-3.32
21	Shekel 5	FDM	4	[0, 10]	-10.1532
22	Shekel 7	FDM	4	[0, 10]	-10.4029
23	Shekel 10	FDM	4	[0, 10]	-10.5364

Next, Figure 2 illustrates the ninety experimental results for the problem of minimizing a multimodal function of Schwefel (ID = 8). It informs that a too-small (10 and 20) or a too-big population p (40 to 100) produces a bad

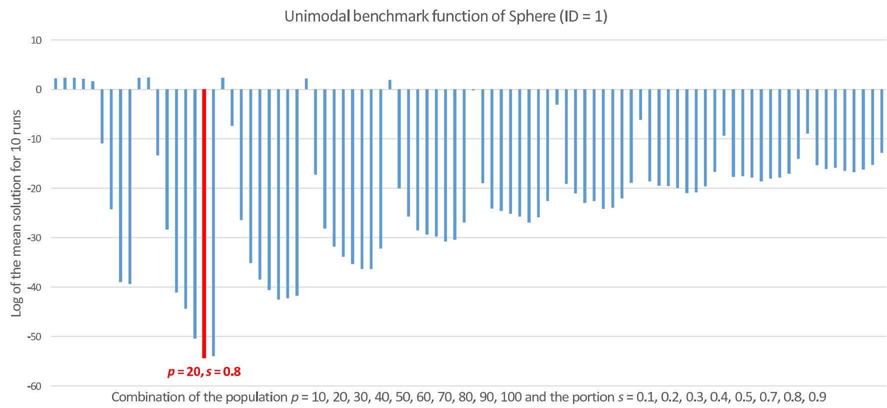


Figure 1: Parameter tuning for a unimodal benchmark function of Sphere (ID = 1)

solution. The bigger the p the worse the solution. A too-big portion s also yields a poor solution. The bigger the s the worse the solution. Therefore, the combination of a too-big p and a too-big s is not recommended. The optimum combination is reached on $p = 30$ and $s = 0.2$. This result proves that a small portion of high-quality individuals in the small population makes the proposed ERA more explorative and faster to find the optimum solution.

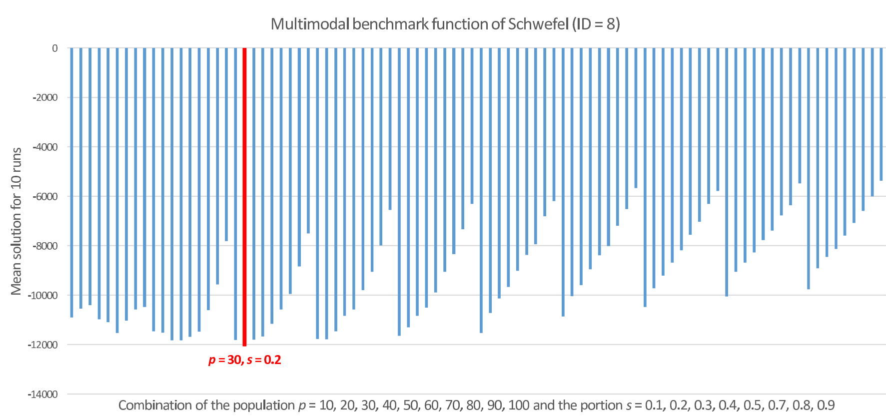


Figure 2: Parameter tuning for a multimodal benchmark function of Schwefel (ID = 8)

Finally, Figure 3 illustrates the ninety experimental results for the problem of minimizing a fixed-dimension multimodal function of Shekel 7 (ID = 22). It shows that a too-small population p (10 to 30) produces a bad solution. The smaller the p the worse the solution. A too-big portion s also yields a poor solution. The bigger the s the worse the solution. Hence, the combination of a too-small p and a too-big s is not recommended. The optimum combination is reached on $p = 100$ and $s = 0.1, 0.2,$ or 0.3. This result informs that a small portion of high-quality individuals in the big population make the proposed ERA balance in the explorative and exploitative searching for an optimum solution.

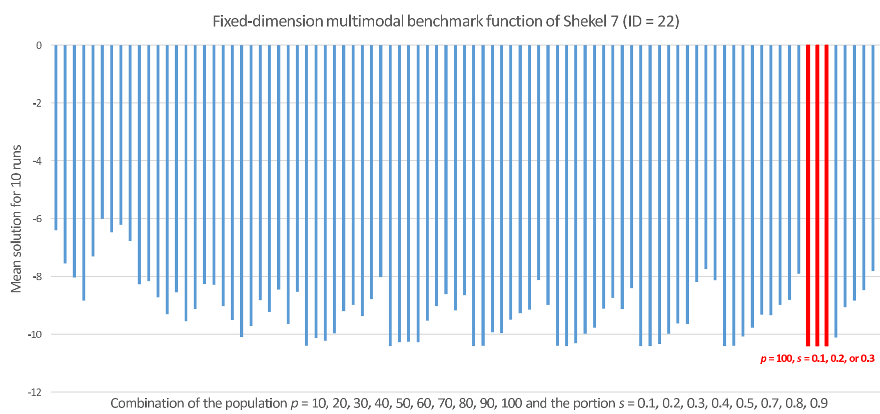


Figure 3: Parameter tuning for a fixed-dimension multimodal benchmark function of Shekel 7 (ID = 22)

3.2. Comparison to the three original Rao algorithms

The proposed ERA is then examined and compared with the three original Rao algorithms: Rao-1, Rao-2, and Rao-3, to search the minimum solutions to the twenty-three benchmark functions listed in Table 1. For each benchmark function, the maximum number of function evaluations is set to 30,000 with 100 runs to reduce the coincidence of the four algorithms. The random seeds of the 100 initial populations (for each benchmark function) are the same when the algorithms use the same population size p to get the fairness. Otherwise, they are different. The Matlab source-code as well as the optimum population sizes used in the Rao-1, Rao-2, and Rao-3 algorithms refer to [29]. Meanwhile, the population size and portion used in ERA are based on the results of the parameter tuning described in subsection 3.1. Table 2 illustrates the examination results based on five metrics: Best solution, Worst solution, Mean solution, standard deviation (STD), and mean function evaluations (MFE), and two optimum parameters of population size and the portion used in each algorithm. The bold text shows the best result while the underscored text informs the second-best (similar) result.

Based on the metric of Worst, Mean, and STD, for the seven unimodal functions, ID = 1 to 7, the proposed ERA mostly outperforms the three Rao algorithms. It achieves much lower mean solutions for the five functions with ID = 1, 2, 3, 4, and 7. It is slightly worse than Rao-1 for the Rosenbrock function (ID = 5), where it gives a mean solution of 31.24156062 while Rao-1 reaches 30.85414709, but it is better than Rao-2 and Rao-3. Unfortunately, it is much worse than Rao-1 for the Step function (ID = 6), where it produces a mean solution of 2.531040325 while Rao-1 obtains 2.32704E-20, but it is better than Rao-2 and Rao-3.

Next, the investigation on the six multimodal functions, ID = 8 to 13, informs that the proposed ERA also mostly outperforms the three Rao algorithms, where it achieves lower mean solutions for the four functions with ID = 8, 9, 10, and 13. It is slightly worse than Rao-1 for the Griewank benchmark function (ID = 11), where it gives a mean solution of 0.025376504 while Rao-1 reaches 0.011749089, but it is better than Rao-2 and Rao-3. It is also slightly worse than Rao-3 for the Penalized benchmark function (ID = 12), where it produces a mean solution of 1.130589565 while Rao-3 obtains 1.099778271, but it is better than Rao-1 and Rao-2.

Finally, the investigation on the metrics of Worst, Mean, and STD for the ten fixed-dimension unimodal functions, ID = 14 to 23, shows that the proposed ERA mostly outperforms the three Rao algorithms, where it achieves lower mean solutions for the six benchmark functions with ID = 15, 16, 19, 20, 21 and 22. It is slightly worse than Rao-2 for the benchmark function of GoldStein-Price (ID = 18), where it gives a mean solution of 3.000302252 while Rao-1 reaches 3, but it is better than Rao-1 and Rao-3. It is also slightly worse than Rao-3 and Rao-2 for the benchmark function of Shekel 10 (ID = 23), where it produces a mean solution of -10.11445995 while Rao-3 and Rao-2 obtain the lower solutions of -10.3742565 and -10.35898843, respectively. Interestingly, based on the metric of Best, it gives a better solution (-10.53644209) than Rao-1 (-10.53644036), Rao-2 (-10.53643886), and Rao-3 (-10.53643734).

Table 2 Comparison of Rao-1, Rao-2, Rao-3, and ERA for 23 benchmark functions

ID	**Metric**	**Rao-1**	**Rao-2**	**Rao-3**	**ERA**
1	Best	7.626E-25	2.12776E-16	1.92673E-51	**8.9808E-63**
	Worst	1.25685E-19	1.6654E-09	2.5956E-40	**9.88121E-53**
	Mean	4.99854E-21	4.9006E-11	9.2457E-42	**2.07426E-54**
	STD	1.73159E-20	2.02461E-10	3.77511E-41	**1.12408E-53**
	MFE	30000	30000	30000	30000
	Population	10	10	10	20
	Portion				0.9
2	Best	3.80459E-16	0.003292845	6.32167E-20	**1.92812E-32**
	Worst	3.99559E-11	10.00763655	1.49865E-13	**1.10184E-26**
	Mean	1.26205E-12	0.121315726	2.08066E-15	**3.26013E-28**
	STD	4.26637E-12	0.998697292	1.59554E-14	**1.5349E-27**
	MFE	30000	30000	30000	30000
	Population	10	20	20	20
	Portion				0.9
3	Best	2.09545E-24	4.20776E-16	8.83731E-34	**2.02601E-60**
	Worst	2.51646E-17	20000	3.1512E-26	**3.73135E-51**
	Mean	2.99381E-19	600	4.48321E-28	**9.35662E-53**
	STD	2.51501E-18	2777.979791	3.19147E-27	**5.15425E-52**
	MFE	30000	30000	30000	30000
	Population	10	10	20	20
	Portion				0.9
4	Best	0.504236253	4.187211667	0.003290339	**6.78005E-06**
	Worst	5.098884898	41.66275728	0.873189273	**0.02903511**
	Mean	2.31254633	16.9361541	0.120113549	**0.001111361**
	STD	1.051753402	7.94162623	0.173710577	**0.003242235**
	MFE	30000	30000	30000	30000
	Population	30	20	20	60
	Portion				0.9
5	Best	0.3084995487	**0.0004289297068**	0.007647765593	0.03847405641

Table 2 Comparison of Rao-1, Rao-2, Rao-3, and ERA for 23 benchmark functions

ID	**Metric**	**Rao-1**	**Rao-2**	**Rao-3**	**ERA**
	Worst	184.9937562	3037.560956	542.6851381	**167.9000192**
	Mean	**34.84929283**	70.49915266	43.36710061	36.48401936
	STD	**31.99209523**	424.8235345	62.41267725	40.6703416
	MFE	30000	30000	30000	30000
	Population	20	10	20	100
	Portion				0.5
6	Best	**3.95396E-25**	1.62207E-12	1.950933713	1.254112212
	Worst	**1.93674E-18**	10100.25	4.760217938	5.935674551
	Mean	**2.32704E-20**	101.0025014	2.978025617	2.531040325
	STD	**1.94378E-19**	1010.025	0.488527412	0.807383928
	MFE	30000	30000	30000	30000
	Population	10	10	30	50
	Portion				0.5
7	Best	0.016777689	0.028244196	0.00313442	**0.002960331**
	Worst	0.160355414	0.263789812	0.041554724	**0.038043669**
	Mean	0.073677235	0.098848861	0.01601691	**0.012131037**
	STD	0.029293298	0.044897583	0.008654207	**0.005647329**
	MFE	30000	30000	30000	30000
	Population	20	20	30	50
	Portion				0.5
8	Best	-10255.35528	-12016.76892	-11641.65114	**-12455.21563**
	Worst	-4304.560919	-5496.871369	-4671.432666	**-11099.67265**
	Mean	-8582.260521	-8673.104501	-9456.600916	**-12015.67076**
	STD	1581.559748	1705.804215	1663.472251	**302.2659241**
	MFE	30000	30000	30000	30000
	Population	10	10	20	30
	Portion				0.2
9	Best	42.78320419	112.8286649	**24.87398785**	26.03169827
	Worst	280.7041958	304.7799026	202.3310204	**51.67285374**
	Mean	99.00834905	192.2113213	100.610812	**37.22129318**
	STD	43.05401847	43.39244946	40.37929687	**5.767324167**
	MFE	30000	30000	30000	30000
	Population	10	10	10	50
	Portion				0.5
10	Best	0.072508061	0.009364202	2.93746E-07	**2.42556E-09**
	Worst	19.96299615	19.9621694	3.21062E-05	**1.00445E-07**

Table 2 Comparison of Rao-1, Rao-2, Rao-3, and ERA for 23 benchmark functions

ID	**Metric**	**Rao-1**	**Rao-2**	**Rao-3**	**ERA**
	Mean	2.56273516	5.14019373	4.68148E-06	**2.29034E-08**
	STD	5.850043995	8.346895116	5.33867E-06	**1.78893E-08**
	MFE	30000	30000	30000	**29970**
	Population	40	20	50	90
	Portion				0.9
11	Best	1.59317E-13	8.88178E-16	**0**	**0**
	Worst	**0.053866937**	0.889568918	0.383999931	0.199815973
	Mean	**0.011749089**	0.09161055	0.030794015	0.025376504
	STD	**0.012163204**	0.187689477	0.050516984	0.03595075
	MFE	30000	30000	27620.4	**26624.8**
	Population	20	10	20	40
	Portion				0.8
12	Best	**6.1519E-12**	0.203655788	0.287826023	0.148850139
	Worst	15.59917497	27.66035428	7.208965188	**5.597209637**
	Mean	1.497987926	7.166688919	**1.099778271**	1.130589565
	STD	3.347109796	5.313811456	1.041309443	**1.011324944**
	MFE	30000	30000	30000	30000
	Population	20	20	50	100
	Portion				0.9
13	Best	1.38482E-06	9.78909E-13	**5.77121E-14**	3.40019E-12
	Worst	10.06545402	42.72956693	0.397445015	**0.098882649**
	Mean	0.456150842	1.51287029	0.020958335	**0.007193028**
	STD	1.638895617	5.048569679	0.052615311	**0.017410352**
	MFE	30000	30000	30000	30000
	Population	30	10	50	80
	Portion				0.8
14	Best	**0.998**	**0.998**	**0.998**	**0.998**
	Worst	**0.998**	4.950491232	1.000298841	**0.998**
	Mean	**0.998**	1.07706363	0.998153122	**0.998**
	STD	**0**	0.556134459	0.000394433	**0**
	MFE	**4144.8**	6682.2	18503	10516.8
	Population	20	20	50	80
	Portion				0.8
15	Best	0.026536947	0.002623873	0.003833459	**0.001984456**
	Worst	0.061428871	0.055006212	0.038766825	**0.036204266**
	Mean	0.042651343	0.033239833	0.027682962	**0.009483111**

Table 2 Comparison of Rao-1, Rao-2, Rao-3, and ERA for 23 benchmark functions

ID	**Metric**	**Rao-1**	**Rao-2**	**Rao-3**	**ERA**
	STD	0.009650198	0.011027106	0.013129211	**0.00705608**
	MFE	30000	30000	30000	30000
	Population	100	20	30	10
	Portion				0.7
16	Best	-1.031628397	-1.03162845	-1.03162827	**-1.031628398**
	Worst	-1.031596371	-0.215463824	-0.215460511	**-1.031600618**
	Mean	-1.031613381	-0.990805801	-1.02344958	**-1.031615353**
	STD	9.76826E-06	0.178771753	0.081615058	**8.44531E-06**
	MFE	2529.7	2338.2	1879.05	**484.6**
	Population	10	5	5	10
	Portion				0.7
17	Best	**0.397887358**	**0.397887358**	**0.397887358**	**0.397887358**
	Worst	**0.397887358**	**0.397887358**	**0.397887358**	**0.397887358**
	Mean	**0.397887358**	**0.397887358**	**0.397887358**	**0.397887358**
	STD	**1.06003E-15**	**1.06003E-15**	**1.06003E-15**	**1.06003E-15**
	MFE	30000	30000	30000	30000
	Population	10	10	10	10
	Portion				0.8
18	Best	**3**	**3**	3.000000356	**3**
	Worst	84	**3**	84	3.024296865
	Mean	3.81	**3**	4.620069482	3.000302252
	STD	8.1	**0**	11.39711885	0.002478975
	MFE	**1447.4**	7089.2	30000	30000
	Population	10	20	10	20
	Portion				0.9
19	Best	-3.862560942	-3.862520503	-3.862708631	**-3.862719148**
	Worst	-1.000816864	-3.860005413	-3.860025992	**-3.8600793**
	Mean	-3.669444166	-3.861179	-3.861199721	**-3.861212482**
	STD	0.594599578	**0.00071381**	0.000765404	0.00076365
	MFE	4311.65	**366.8**	506.7	403.6
	Population	5	20	30	20
	Portion				0.7
20	Best	-3.321897003	-3.321588453	-3.321846058	**-3.32196047**
	Worst	-3.173734332	-1.709685086	**-3.203161918**	-3.20311344
	Mean	-3.254373419	-3.240408817	-3.266467027	**-3.281656468**
	STD	0.056445682	0.182074904	0.058723738	**0.055372259**

Table 2 Comparison of Rao-1, Rao-2, Rao-3, and ERA for 23 benchmark functions

ID	Metric	Rao-1	Rao-2	Rao-3	ERA
	MFE	19015.2	13126.3	14504.7	**10539**
	Population	20	10	30	10
	Portion				0.2
21	Best	-10.15319786	-10.15319944	-10.15319871	**-10.15319966**
	Worst	-2.625619	-2.630471668	-2.630471668	**-5.057296338**
	Mean	-6.859391671	-7.257911109	-7.372318656	**-10.07891808**
	STD	2.052147876	3.052405481	2.82458379	**0.513608708**
	MFE	28134.2	**16867.2**	21307.2	17631
	Population	20	20	30	100
	Portion				0.2
22	Best	**-10.40291414**	-10.40291397	-10.40291287	-10.40291383
	Worst	-2.738399204	-2.74956211	-4.803725772	**-10.07051811**
	Mean	-8.047697928	-9.811363404	-9.372551375	**-10.38018796**
	STD	2.624281948	1.517964991	1.188005718	**0.064089416**
	MFE	22891	**14785.5**	26734	19980
	Population	20	50	100	100
	Portion				0.2
23	Best	-10.53644036	-10.53643886	-10.53643734	**-10.53644209**
	Worst	-2.790290505	**-7.947019775**	-4.49818821	-4.069876498
	Mean	-9.57173758	-10.35898843	**-10.3742565**	-10.11445995
	STD	1.937147518	**0.4798249682**	0.8449893664	1.448143483
	MFE	14279.4	17459	**8475**	16547
	Population	20	100	50	100
	Portion				0.1

Based on the results in Table 2, the performance of the proposed ERA in terms of the number of functions where it gives better, equal, and worse solutions than the original Rao algorithms is summarized in Table 3. For the metrics of Worst, Mean, and STD, the proposed ERA produces better solutions for fifteen benchmark functions with ID: 1, 2, 3, 4, 7, 8, 9, 10, 13, 15, 16, 19, 20, 21, and 22. It obtains equal solutions for two benchmark functions with ID: 14 and 17. It produces the worse solutions for six benchmark functions, where it obtains slightly worse results for five functions with ID: 5, 11, 12, 18, and 23, but it gives a much worse solution only for one function (ID = 6). These results prove that the proposed ERA is better and more stable than the original Rao algorithms to solve three kinds of benchmark functions: unimodal, multimodal, and fixed-dimension multimodal.

For the metric of Best, the ERA achieves the better solutions for twelve benchmark functions with ID: 1, 2, 3, 4, 7, 8, 10, 13, 15, 19, 20, and 23. It obtains equal results for four functions with ID: 11, 14, 17, and 18. It gives the worse solutions for seven functions with ID: 5, 6, 9, 12, 16, 21, and 22. Meanwhile, for the metric of MFE, all the four algorithms give the same results for thirteen functions and similar achievements for other functions. The ERA is better than three Rao algorithms for the four benchmark functions with ID: 10, 11, 16, and 20. Especially, for the function ID = 16, it significantly gives a lower MFE than the others.

Table 3: Performance of the proposed ERA in terms of the number of functions where it gives better, equal, and worse solutions than the original Rao algorithms (Rao-1, Rao-2, and Rao-3)

Metric	Better	Equal	Worse
Best	12	4	7
Worst	15	2	6
Mean	15	2	6
STD	15	2	6
MFE	4	13	6

3.3. Detailed investigation on unimodal functions

A detailed investigation of the seven benchmark unimodal functions, ID = 1 to 7, is discussed by illustrating some convergence analysis of the proposed ERA and the original Rao algorithms. For each benchmark function, the maximum number of function evaluations is set to 30,000 with 100 runs to reduce the coincidence of the four algorithms. Figure 4 shows the evolution of all the algorithms until convergence to the optimum solution for the benchmark function of Sphere (ID = 1). The horizontal axis is the generation, which is calculated as 30,000 function evaluations divided by the population size p. The random seeds of the 100 initial populations are the same for the algorithms that use the same optimum population size p. Hence, in this case, the three original Rao algorithms use the same initial population since they have the same optimum $p = 10$. In contrast, the ERA uses a different initial population because it has the optimum $p = 20$. Due to the different optimum p for each algorithm, then the evolution is illustrated using the different step size of generation to get the fairness. Here, the proposed ERA uses a step size of 1 while the three original Rao algorithms use a step size of 2 so that all the algorithms show the same generations of 1 to 1,500. It can be seen in Figure 4 that the ERA is the fastest algorithm, where it can converge at the beginning of the evolution. This result also applies to four other unimodal functions ID = 2, 3, 4, and 7.

Figure 5 shows the evolution of all the algorithms for the benchmark function of Schwefel 2.21 (ID = 4). It can be seen that the ERA is much faster than the others, where it converges on the generation of 250 (half of the evolution). At the end of evolution, it gives the lowest mean solution of 0.001111361 while Rao-1, Rao-2, and Rao-3 produce worse solutions of 2.31254633, 16.9361541, and 0.120113549, respectively. This result informs that the ERA is quite fast to converge to the global solution because of its exploitative ability created by the big population of high-quality individuals and the crossover applied to the fittest individual X_{Best}.

Next, the convergence analysis is performed for the two unimodal functions with ID = 5 and 6, where ERA gives the worse solutions. Figure 6 shows that at the beginning of evolution (generation 1 to 50), the ERA converges more slowly than both Rao-3 and Rao-1 in minimizing the Rosenbrock function (ID = 5). Interestingly, it overtakes the Rao-3 that is getting stuck at generation 240 and then keeps evolves to converge to the solution that very close to the Rao-1 at the last generation (300), as illustrated in Figure 7. This result indicates that the ERA is not easy to be trapped on the local solution because of its explorative

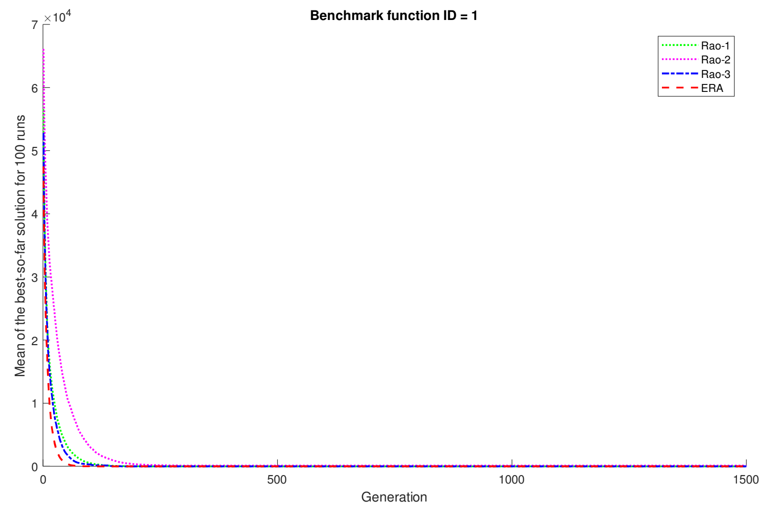


Figure 4: Convergence analysis for a unimodal benchmark function of Sphere (ID = 1)

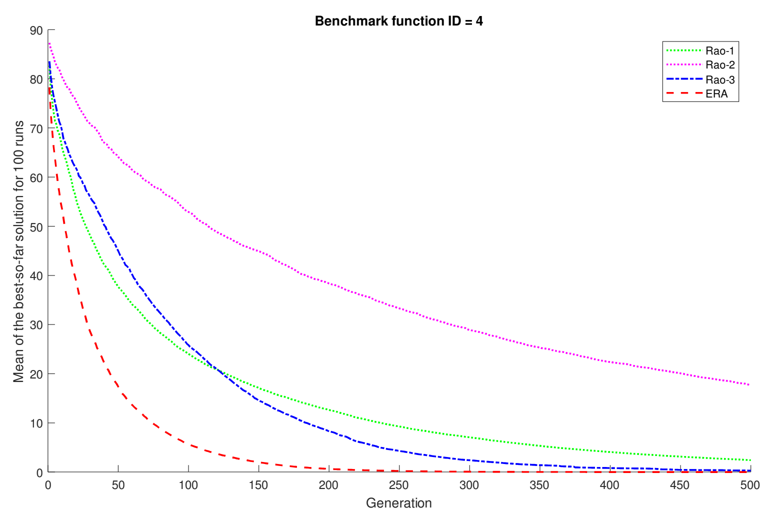


Figure 5: Convergence analysis for a unimodal benchmark function of Schwefel 2.21 (ID = 4)

ability created by the big sub-population of the low-quality individuals that do a new random walk and also the mutation operator performed to the fittest individual X_{Best}. A similar result applies to the Step function (ID = 6).

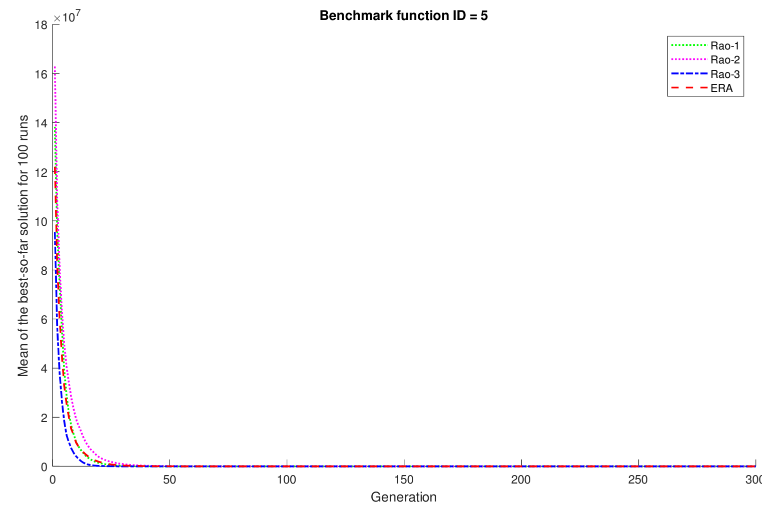


Figure 6: Convergence analysis for a unimodal benchmark function of Rosenbrock (ID = 5)

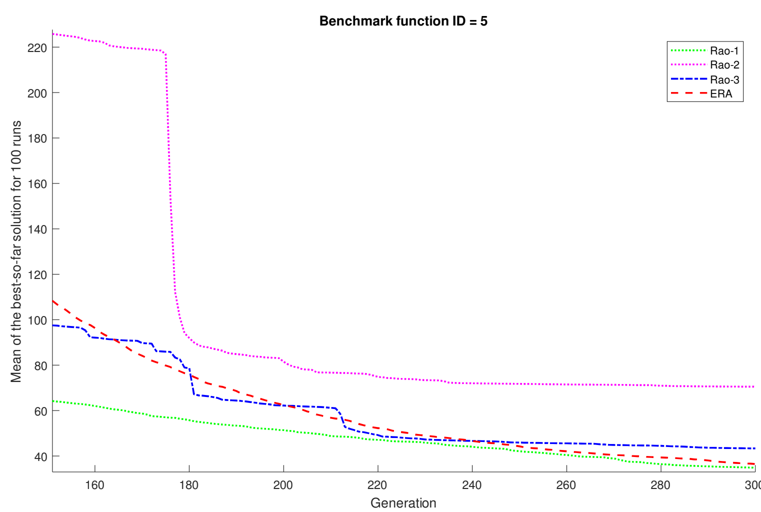


Figure 7: Convergence analysis for a unimodal benchmark function of Rosenbrock (ID = 5) for the generation of 151 to 300

3.4. Detailed investigation on multimodal functions

Next, a detailed investigation of the seven benchmark multimodal functions, ID = 8 to 13, is illustrated by some convergence analysis of the proposed ERA and the three original Rao algorithms. Figure 8 shows the evolution of all the algorithms for the benchmark function of Schwefel (ID = 8). The ERA performs the best evolution and converges to a much better solution than the others. It seems to evolve more slowly than the Rao-2 algorithm at the beginning of the evolution (generation 1 to 100), but it evolves much faster at generation 101 to 1,000 and finally converges to a much better solution than the others. This result also applies to three other multimodal functions ID = 9, 10, and 13.

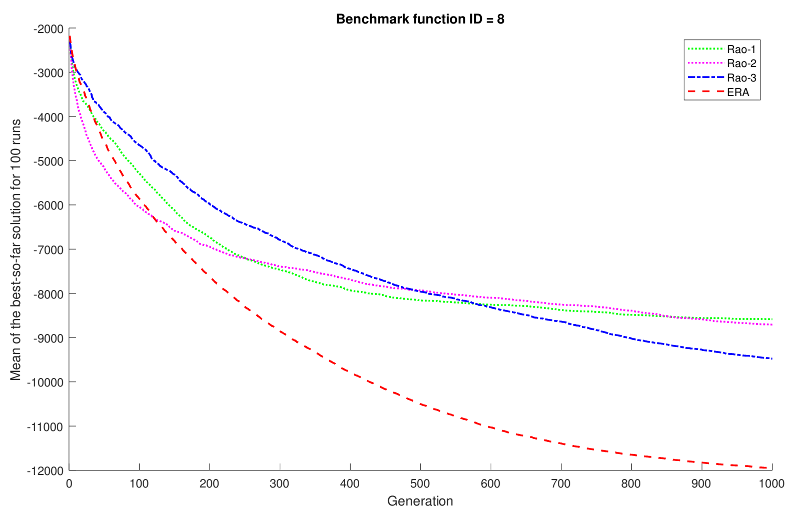


Figure 8: Convergence analysis for a multimodal benchmark function of Schwefel (ID = 8)

Next, the convergence analysis is given for a multimodal Griewank function with ID = 11, where ERA gives a worse solution. Figure 9 shows that the ERA converges much faster than the others at the beginning of evolution: generation 1 to 50. It works slower than the Rao-1 on generation 375 (half evolution) and converges to the slightly worse solution on the last generation 750, as illustrated in Figure 10. This result implies that the ERA can be trapped on the local solution when it is too-exploitative created by the too-small sub-population of the low-quality individuals.

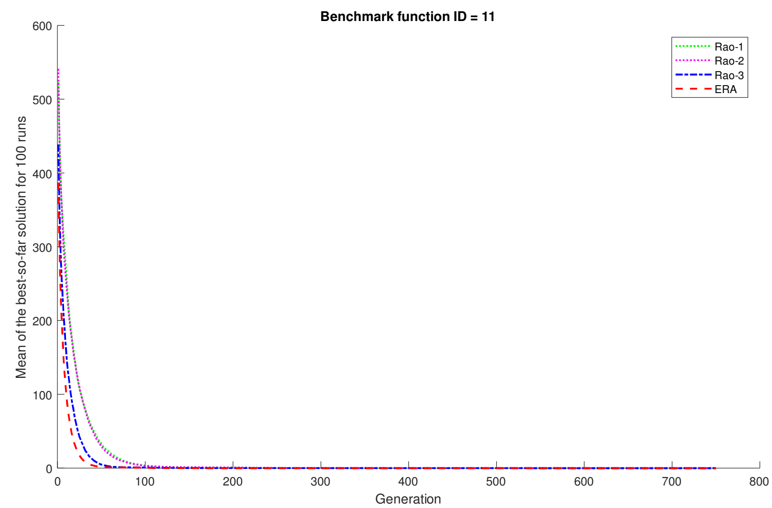


Figure 9: Convergence analysis for a multimodal benchmark function of Griewank (ID = 11)

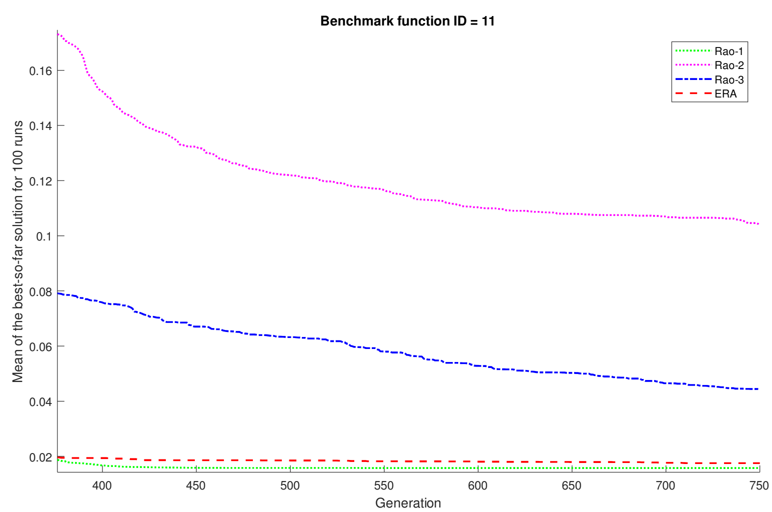


Figure 10: Convergence analysis for a multimodal benchmark function of Griewank (ID = 11) for the generation 376 to 750

3.5. Detailed investigation on fixed-dimension multimodal functions

Finally, a detailed investigation of ten benchmark fixed-dimension unimodal functions, ID = 14 to 23, is also illustrated by some convergence analysis of the proposed ERA and the original Rao algorithms. Figure 11 shows the evolution of all the algorithms for the benchmark function of Shekel 5 (ID = 21). The ERA performs the best evolution and converges to a much better solution than the others. It seems to evolve more slowly than the Rao-1, Rao-2, and Rao-3 algorithms at the beginning of the evolution (generation 1 to 30), but it evolves much faster at generation 31 to 50 and keeps evolving until finally converges to a much better solution than the others. This result also applies to seven other multimodal functions with ID = 14, 15, 16, 17, 19, 20, and 22.

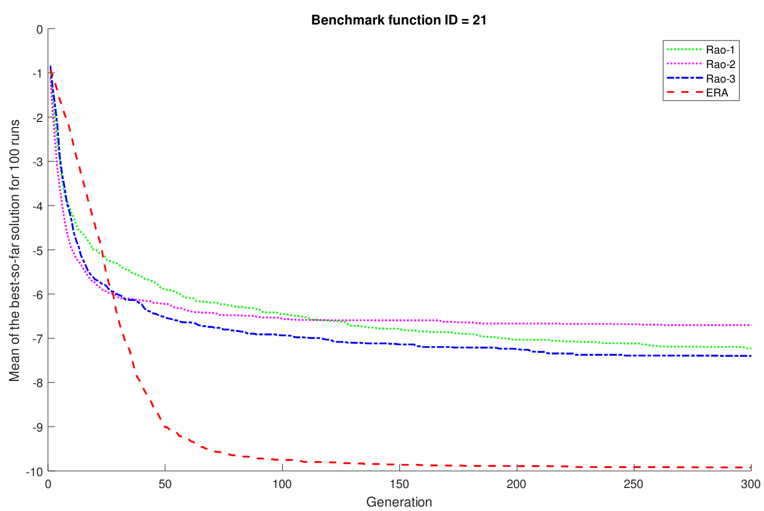


Figure 11: Convergence analysis for a fixed-dimension multimodal benchmark function of Shekel 5 (ID = 21)

Furthermore, the convergence analysis is carried out for a fixed-dimension multimodal Shekel function with ID = 23, where ERA gives a slightly worse solution. Figure 12 shows that at the beginning of the evolution, generation 1 to 125, the ERA evolves similarly to both Rao-3 and Rao-2 algorithms. It is getting stuck at generation 126 and converges to a slightly worse solution at the last generation 300. This result shows that the ERA can be trapped on the local solution when it works in a high-explorative manner created by the big sub-population of the low-quality individuals.

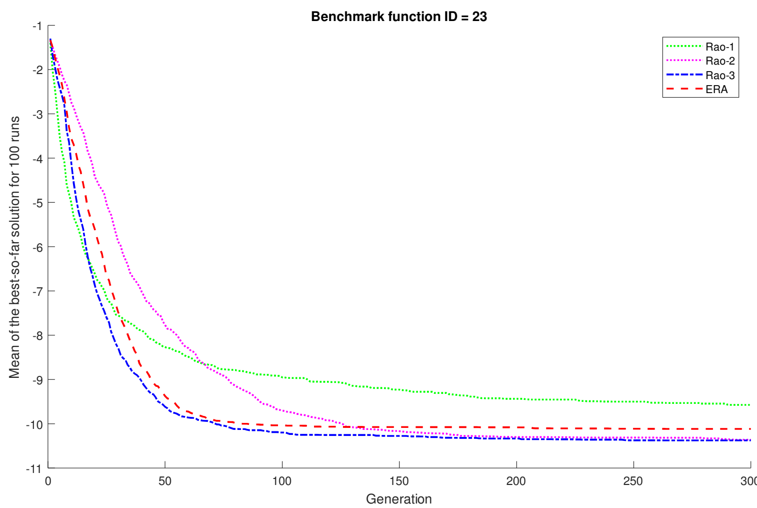


Figure 12: Convergence analysis for a fixed-dimension multimodal benchmark function of Shekel 10 (ID = 23)

4. Conclusion

The proposed evolutionary Rao algorithm (ERA) works very well based on two additional schemes: splitting the population into two sub-populations based on their qualities: high and low, with a proper portion, and exploiting two evolutionary operators: crossover and mutation. Evaluation of the twenty-three benchmark functions shows that it outperforms three original Rao algorithms, where it gives better mean solutions for fifteen functions: five of the seven unimodal functions, four of the six multimodal functions, and six of the ten fixed-dimension multimodal functions. It obtains the same average solutions for two fixed-dimension multimodal functions. It just gives slightly worse solutions for six functions: two of the seven unimodal functions, two of the six multimodal functions, and two of the ten fixed-dimension multimodal functions. A detailed investigation informs that both introduced schemes work well as they are designed to make the ERA keeps evolving until the end of evolution and avoid being trapped on the local optimum solutions. In the future, a comprehensive examination as well as an application development can be performed to see its performance in handling some real world problems.

References

[1] K. Hussain, M. N. Mohd Salleh, S. Cheng, Y. Shi, Metaheuristic research: a comprehensive survey, Artificial Intelligence Review 52 (4) (2019) 2191–2233. doi:10.1007/s10462-017-9605-z.
URL https://link.springer.com/article/10.1007/s10462-017-9605-z

[2] J. H. Holland, Genetic algorithms, Scientific American 267 (1) (1992) 66–72. doi:10.1038/scientificamerican0792-66.
URL https://www.scientificamerican.com/article/genetic-algorithms/

[3] A. Lambora, K. Gupta, K. Chopra, Genetic Algorithm- A Literature Review, in: 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), 2019, pp. 380–384. doi:10.1109/COMITCon.2019.8862255.

[4] F. Murillo, T. Neuenschwander, R. Dornberger, T. Hanne, Optimization of a Robotic Manipulation Path by an Evolution Strategy and Particle Swarm Optimization, in: ACM International Conference Proceeding Series, Association for Computing Machinery, 2020, pp. 36–41. doi:10.1145/3396474.3396488.

[5] Z. Li, Q. Zhang, Variable metric evolution strategies by mutation matrix adaptation, Information Sciences 541 (2020) 136–151. doi:10.1016/j.

ins.2020.05.091.
URL https://doi.org/10.1016/j.ins.2020.05.091

[6] J. Cheng, Z. Pan, H. Liang, Z. Gao, J. Gao, Differential evolution algorithm with fitness and diversity ranking-based mutation operator, Swarm and Evolutionary Computation 61 (May 2020) (2021) 100816. doi:10.1016/j.swevo.2020.100816.
URL https://doi.org/10.1016/j.swevo.2020.100816

[7] R. Zhang, Z. Qiu, Optimizing hyper-parameters of neural networks with swarm intelligence: A novel framework for credit scoring, PLoS ONE 15 (6). doi:10.1371/journal.pone.0234254.

[8] V. Mp, B. Anand, Microprocessors and Microsystems Particle swarm optimization technique for multilevel inverters in solar harvesting micro grid system, Microprocessors and Microsystems 79 (August) (2020) 103288. doi:10.1016/j.micpro.2020.103288.
URL https://doi.org/10.1016/j.micpro.2020.103288

[9] D. Kumar, B. G. R. Gandhi, R. K. Bhattacharjya, Firefly Algorithm and Its Applications in Engineering Optimization, Modeling and Optimization in Science and Technologies 16 (2020) 93–103. doi:10.1007/978-3-030-26458-1_6.
URL https://www.springerprofessional.de/en/firefly-algorithm-and-its-applications-in-engineering-optimizati/17575198

[10] V. Kumar, D. Kumar, A Systematic Review on Firefly Algorithm: Past, Present, and Future, Archives of Computational Methods in Engineeringdoi:10.1007/s11831-020-09498-y.

[11] S. Gupta, K. Deep, Enhanced leadership-inspired grey wolf optimizer for global optimization problems, Engineering with Computers 36 (4) (2020) 1777–1800. doi:10.1007/s00366-019-00795-0.

[12] J. C. Bansal, S. Singh, A better exploration strategy in Grey Wolf Optimizer, Journal of Ambient Intelligence and Humanized Computingdoi:10.1007/s12652-020-02153-1.

[13] A. S. Assiri, A. G. Hussien, M. Amin, Ant lion optimization: Variants, hybrids, and applications, IEEE Access 8 (2020) 77746–77764. doi:10.1109/ACCESS.2020.2990338.
URL https://ieeexplore.ieee.org/abstract/document/9078091

[14] J. H. Holland, J. H. Holland, Others, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence, University of Michigan press, 1975.

[15] A. C. Rizal, S. Suyanto, Human-Like Constrained-Mating to Make Genetic Algorithm More Explorative, in: 2020 8th International Conference on Information and Communication Technology (ICoICT), IEEE, 2020, pp. 1–5. doi:https://doi.org/10.1109/ICoICT49345.2020.9166387.
URL https://ieeexplore.ieee.org/document/9166387/

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

[17] D. Li, W. Guo, A. Lerch, Y. Li, L. Wang, Q. Wu, An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization, Swarm and Evolutionary Computation 60 (December 2019) (2021) 100789. doi:10.1016/j.swevo.2020.100789.
URL https://doi.org/10.1016/j.swevo.2020.100789

[18] D. Sedighizadeh, E. Masehian, M. Sedighizadeh, H. Akbaripour, ScienceDirect GEPSO : A new generalized particle swarm optimization algorithm, Mathematics and Computers in Simulation 179 (2021) 194–212. doi:10.1016/j.matcom.2020.08.013.
URL https://doi.org/10.1016/j.matcom.2020.08.013

[19] X.-S. Yang, Firefly algorithms for multimodal optimization, in: Stochastic Algorithms: Foundations and Applications, Springer Berlin Heidelberg, 2009, pp. 169–178.

[20] I. Fister, I. Fister, X.-S. Yang, J. Brest, A comprehensive review of firefly algorithms, Swarm and Evolutionary Computation 13 (2013) 34–46.

[21] H. Peng, W. Zhu, C. Deng, Z. Wu, Enhancing firefly algorithm with courtship learning, Information Sciences 543 (2020) 18–42. doi:10.1016/j.ins.2020.05.111.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85088912963{&}doi=10.1016{%}2Fj.ins.2020.05.111{&}partnerID=40{&}md5=96db75459ce8599098979d9c83af6772

[22] S. Wu, Z. Wu, H. Peng, Enhancing Firefly Algorithm with Best Neighbor Guided Search Strategy, Wuhan University Journal of Natural Sciences 24 (6) (2019) 524–536. doi:10.1007/s11859-019-1432-4.

[23] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey Wolf Optimizer, Advances in Engineering Software 69 (2014) 46–61. doi:10.1016/j.advengsoft.2013.12.007.

[24] J. Luo, Z. Liu, Novel grey wolf optimization based on modified differential evolution for numerical function optimization, Applied Intelligence 50 (2) (2020) 468–486. doi:10.1007/s10489-019-01521-5.
URL https://link.springer.com/article/10.1007/s10489-019-01521-5

[25] S. Bahuguna, A. Pal, Annealed grey wolf optimization, Advances in Mathematics: Scientific Journal 9 (8) (2020) 5477–5489. doi:10.37418/amsj.9.8.18.

[26] W. Long, T. Wu, S. Cai, X. Liang, J. Jiao, M. Xu, A Novel Grey Wolf Optimizer Algorithm with Refraction Learning, IEEE Access 7 (2019) 57805–

57819. doi:10.1109/ACCESS.2019.2910813. URL https://ieeexplore.ieee.org/document/8689025

[27] M. H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, An improved grey wolf optimizer for solving engineering problems, Expert Systems with Applications 166. doi:10.1016/j.eswa.2020.113917. URL https://www.sciencedirect.com/science/article/pii/S0957417420307107

[28] S. Mirjalili, Advances in Engineering Software The Ant Lion Optimizer, Advances in Engineering Software 83 (2015) 80–98. doi:10.1016/j.advengsoft.2015.01.010. URL http://dx.doi.org/10.1016/j.advengsoft.2015.01.010

[29] R. V. Rao, Rao algorithms: Three metaphor-less simple algorithms for solving optimization problems, International Journal of Industrial Engineering Computations 11 (1) (2020) 107–130. doi:10.5267/j.ijiec.2019.6.002.

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Evidence of correspondence

Evolutionary Rao Algorithm

1. First submission (04 January 2021)
2. **LoA with Major Revision (23 February 2020)**
3. Responses to Reviewers, Final submission (12 March 2021)
4. LoA with Fully Accepted (04 April 2021)
5. Proof Reading (11 April 2021)

Telkom University

SUYANTO SUYANTO <suyanto@telkomuniversity.ac.id>

Decision on submission to Journal of Computational Science

1 message

Journal of Computational Science <em@editorialmanager.com> Tue, Feb 23, 2021 at 10:57 PM
Reply-To: Journal of Computational Science <jocs@elsevier.com>
To: Suyanto Suyanto <suyanto@telkomuniversity.ac.id>

Manuscript Number: JOCSCI-D-21-00030

Evolutionary Rao Algorithm

Dear Dr. Suyanto,

Thank you for submitting your manuscript to Journal of Computational Science.

I have completed my evaluation of your manuscript. The reviewers recommend reconsideration of your manuscript following major revision. I invite you to resubmit your manuscript after addressing the comments below. Please resubmit your revised manuscript by Apr 24, 2021.

When revising your manuscript, please consider all issues mentioned in the reviewers' comments carefully: please outline every change made in response to their comments and provide suitable rebuttals for any comments not addressed. Please note that your revised submission may need to be re-reviewed.

To submit your revised manuscript, please log in as an author at https://www.editorialmanager.com/jocsci/, and navigate to the "Submissions Needing Revision" folder.

Journal of Computational Science values your contribution and I look forward to receiving your revised manuscript.

Kind regards,
Valeria Krzhizhanovskaya
Editor-in-Chief

Journal of Computational Science

Editor and Reviewer comments:

Reviewer #1

The authors present an Evolutionary Rao Algorithm (ERA) obtained by improving a metaheuristic Rao algorithm. Inspired by a Firefly Algorithm, the first improvement consists in splitting the population into a high-quality population and a low-quality population according to a proportion defined by the user. The proportion of split controls the degree of exploitation and exploration of the algorithm. Inspired by Genetic Algorithms, the second improvement consists in applying a crossover and a mutation operator to the best individual and the best pretender (obtained by moving the high-quality population towards the best individual).

The article is well written, references are up-to-date, the motivations for the improvements are well justified and the experiments are sound (extensive hyper-parameter grid search, high number of experiment repetitions, high number of benchmark functions). The key points of the proposed algorithm are the few number of hyper-parameters and its performance on multi-modal benchmark functions. Unfortunately, the proposed algorithm is not applied on a real-world problem and it is not compared to the Firefly Algorithm it is inspired by.

I suggest minor modifications to further improve the manuscript:

1) The proposed algorithm should be applied on a real-world problem. European Space Agency provides Global Trajectory Optimisation Problems implemented in Matlab.

2) A comparison with the Enhanced Firefly Algorithm [21] you are inspired by should be considered. The Matlab code for the Enhanced Firefly Algorithm is available on-line.

3) In Section 2 (line 138 page 6) you states that crossover favors exploitation while mutation favors exploration but in Section 1 (line 17 page 2) you states that crossover and mutation are responsible for exploration. Can you please be clearer?

4) Equation (7) in Section 2.3. (page 9) does not provide any information about the mutation operator employed.

5) The difference between multi-modal functions (ID=8 to 13) and fixed-dimension multi-modal functions (ID=14 to 23) is not clear. Is the difference only about decision space dimensions? In this case, it could be convenient to rename "fixed-dimension multi-modal functions" into "low-dimension multi-modal functions".

6) In Section 3.1 it is stated that the results of 20 other benchmark functions are similar to those of functions 1, 8 and 22 (line 208,

page 10). According to Table 2, it seems not to be the case. Indeed for function ID=8 population size is 30 and proportion is 0.2 while for function ID=10 population size is 90 and proportion is 0.9.

7) According to Figure 2, a low population size seems to be better to treat the 30-d Schwefel problem. According to Figure 3, a high population size seems to be better to treat the Shekel 4-d problem. I found counterintuitive that a lower population size is preferred to treat a more difficult problem (my first guess is a 30-d problem is more difficult than a 4-d problem). Could you please give an hypothesis to explain this observation?

8) Typos and sentences that could be rephrased:
- They make GWO has a high exploitative searching strategy. (line 63, page 4)
- for each wolf to enhances the balance (line 70, page 4)
- an ability so solve (line 79, page 4)
- the best candidate as value of variable j (line 93, page 5)
- X_{m,LQ,i} and X_{m,HQ,n} is the (line 181, page 9)
- not all dimensions is used (line 183, page 9)
- a small portion [...] make (line 236, page 13)
- fixed-dimension unimodal functions (line 274, page 14)
- keeps evolves (line 341, page 21)
- to converge to the solution that very close to the Rao-1 (line 341, page 21)
- fixed-dimension unimodal (line 366, page 26)
- keeps evolving until finally converges (line 378, page 26)

Reviewer #2: The paper proposed an improved RAO algorithm called Evolutionary RAO Algorithm (ERA). ERA differs from RAO-1, RAO-2 and RAO-3 in that it has two additional schemes: (a) the first scheme is a population splitting mechanis which divides the main population into two sub-populations based on their fitness. (b) A crossover and mutation operators were also used in the proposed method. Moreover, a random walk was also used.
The performance of ERA has been validated by means of numerical experiments on well-known benchmark functions. Although the manuscript is well-organized and well-written, the following aspects of the paper needs major revisions:
Regarding the test functions:
The used test functions, indeed, are very well-known and widely used benchmark functions in the literature. However, the following points should be handled:
1) The authors should justify the choice of benchmark functions. Majority of the used test functions have symmetric search space boundaries, and in most of them the optimal point x* is in (0,0, …,0). Other test functions should be considered to investigate (a) the bias of the proposed algorithm toward the center of the search space, and (b) the effect of shift and rotation of the test function on the behavior of the proposed method. (c) the effect of noise in the fitness function.
Regarding the simulation results and performance comparison:
2) It is not clear how the authors validated the difference between numerical results of couterpart algorithms. Authors are asked to conduct a non-parametric statistical analysis, i.e. Wilcoxon rank sum test, on the results and report the p-values.
3) When reporting the results of the proposed algorithm, the authors conducted a considerable amount of experiments to find the sub-optimal values for p and s parameters. Although the proposed parameters are beneficial to the performance of the ERA, the used approach to find their best values for each benchcmark function is not fair. Indeed, the process of finding the best values for the two parameters is a huge computational burden which simply neeglected when judging about the superiority of the proposed method. Therefore, the authors should use p and s with the same values for all benchmark functions.
4) The computational complexity of the proposed ERA should be calculated and compared with those of the other algorithms.

Reviewer #3

Manuscript No.: JOCSCI-D-21-00030

Manuscript title: Evolutionary Rao Algorithm

The above manuscript proposed Evolutionary Rao Algorithm and its application for solving unconstrained optimization problems. Several unconstrained benchmarks are considered. The manuscript lacks sufficient contribution and novelty and based on the following major comments, the manuscript is not ready for publication due the following comments:

- The concept of sub-population will add another user parameters, and there will be question here, what is the optimal number of sub-population number? Regarding cross-over and mutation rates, both are considered as user parameters which are important factors. Therefore, user parameters are not considerer as population and s which were fine tuned.

- The benchmarks used are simple and very standard version of benchmarks. Those are not shifted, rotated and hybrid. They are in their simple format. It is strongly suggested to examine their improved method over CEC benchmark series.

- Talking about sensitivity analysis, the reviewer is not convinced with the methodology applied in the manuscript. It is mostly based on try and error various combinations of user parameters. It is strongly suggested to use well-known method such as Taguchi approach to fine tune initial parameters.

- The manuscript only focuses on unconstrained benchmarks which is not acceptable at this moment. Ability of handling constrained benchmarks should be validated.

- Comparison pool is not sufficient and fair. It is intended to include more recent and competitive optimizers not only some variants of RAO. We wish to developed, improved an optimizer to make it better than the existing ones. This is the main purpose and target of improving/developing/hybridizing algorithms.

- Concerning presenting simulation results, there are only statistical results such as mean, SD, worst, and best. There is no report of statistical tests such as T-test and Friedman. How can be sure about the significance of the obtained results?

- Future research is missing. Conclusions instead of conclusion.

- Please double proofread the entire manuscript. It can be seen some typo-mistakes and grammatical errors in the manuscript.

Evidence of correspondence

Evolutionary Rao Algorithm

1. First submission (04 January 2021)
2. LoA with Major Revision (23 February 2021)
3. **Responses to Reviewers, Final submission (12 March 2021)**
4. LoA with Fully Accepted (04 April 2021)
5. Proof Reading (11 April 2021)

Telkom University

SUYANTO SUYANTO <suyanto@telkomuniversity.ac.id>

Confirming submission to Journal of Computational Science

1 message

Journal of Computational Science <em@editorialmanager.com> Fri, Mar 12, 2021 at 10:00 PM
Reply-To: Journal of Computational Science <jocs@elsevier.com>
To: Suyanto Suyanto <suyanto@telkomuniversity.ac.id>

This is an automated message.

Manuscript Number: JOCSCI-D-21-00030R1

Evolutionary Rao Algorithm

Dear Dr. Suyanto,

We have received the above referenced manuscript you submitted to Journal of Computational Science.

To track the status of your manuscript, please log in as an author at https://www.editorialmanager.com/jocsci/, and navigate to the "Revisions Being Processed" folder.

Thank you for submitting your revision to this journal.

Kind regards,
Journal of Computational Science

More information and support

You will find information relevant for you as an author on Elsevier's Author Hub: https://www.elsevier.com/authors

FAQ: How can I reset a forgotten password?
https://service.elsevier.com/app/answers/detail/a_id/28452/supporthub/publishing/
For further assistance, please visit our customer service site: https://service.elsevier.com/app/home/supporthub/publishing/
Here you can search for solutions on a range of topics, find answers to frequently asked questions, and learn more about Editorial Manager via interactive tutorials. You can also talk 24/7 to our customer support team by phone and 24/7 by live chat and email

____

Journal of Computational Science

Evolutionary Rao Algorithm

--Manuscript Draft--

Manuscript Number:	JOCSCI-D-21-00030R1
Article Type:	Full Length Article
Keywords:	evolutionary Rao algorithm; exploitation-exploration balance; fitness-based adaptation scheme; random walk; two subpopulations
Corresponding Author:	Suyanto Suyanto Telkom University Bandung, INDONESIA
First Author:	Suyanto Suyanto
Order of Authors:	Suyanto Suyanto
	Agung Toto Wibowo
	Said Al Faraby
	Siti Sa'adah
	Rita Rismala
Abstract:	This paper proposes an evolutionary Rao algorithm (ERA) to enhance three state-of-the-art metaheuristic Rao algorithms (Rao-1, Rao-2, Rao-3) by introducing two new schemes. Firstly, the population is split into two sub-populations based on their qualities: high and low, with a particular portion. The high-quality sub-population searches for an optimum solution in an exploitative manner using a movement scheme used in the Rao-3 algorithm. Meanwhile, the low-quality one does in an explorative fashion using a new random walk. Secondly, two evolutionary operators: crossover and mutation, are incorporated to provide both exploitation and exploration strategies. A fitness-based adaptation is introduced to dynamically tune the three parameters: the portion of high-quality individuals, mutation radius, and mutation rate throughout the evolution, based on the improvement of best-so-far fitness. In contrast, the crossover is implemented using a standard random scheme. Comprehensive examinations using 38 benchmarks: twenty-three classic functions, ten CEC-C06 2019 benchmarks, and five global trajectory optimization problems show that the proposed ERA generally outperforms the four competitors: Rao-1, Rao-2, Rao-3, and firefly algorithm with courtship learning (FA-CL). Detailed investigations indicate that both proposed schemes work very well to make ERA evolves in an exploitative manner, which is created by a high portion of high-quality individuals and the crossover operator, and avoids being trapped on the local optimum solutions in an explorative manner, which is generated by a high portion of low-quality individuals and the mutation operator. Finally, the adaptation scheme effectively controls the exploitation-exploration balance by dynamically tuning the portion, mutation radius, and mutation rate throughout the evolution process.
Suggested Reviewers:	Tiebin Wu wutiebin81@csu.edu.cn Tiebin Wu is doing some researches about metaheuristic algorithms.
	Hu Peng hu_peng@whu.edu.cn Hu Peng is interested in the swarm intelligences area.
Opposed Reviewers:	
Response to Reviewers:	

March 11, 2021

Dear Peter Sloot,

I wish to submit the revised full manuscript entitled “Evolutionary Rao Algorithm”. In this revised manuscript, all comments and suggestions given by the reviewers are carefully addressed (yellow highlight). However, due to many additional explanations, the revised manuscript is now 50 pages. Furthermore, this revised manuscript has been checked using both Grammarly Premium and iThenticate with a low similarity index of 16%, without exclude any source.

Thank you for your consideration of this manuscript. Please address all correspondence concerning this manuscript to me at suyanto@telkomuniversity.ac.id.

Sincerely,

Suyanto
Associate Professor at Telkom University
Jl. Telekomunikasi Terusan Buah Batu Bandung 40257, Indonesia

Authors' Responses to Reviewers' Comments

Dear Reviewers,

Thank you very much for your comments and suggestions that helped us to prepare a hopefully better version of our manuscript. Below are our responses and corrections to the comments and suggestions, where the blue texts are our responses, the purple ones are the original text in the manuscript, ~~the red strikethrough ones~~ are the text "to be deleted", and the green ones are the text "to be inserted".

Reviewer #1

The authors present an Evolutionary Rao Algorithm (ERA) obtained by improving a metaheuristic Rao algorithm. Inspired by a Firefly Algorithm, the first improvement consists in splitting the population into a high-quality population and a low-quality population according to a proportion defined by the user. The proportion of split controls the degree of exploitation and exploration of the algorithm. Inspired by Genetic Algorithms, the second improvement consists in applying a crossover and a mutation operator to the best individual and the best pretender (obtained by moving the high-quality population towards the best individual).

The article is well written, references are up-to-date, the motivations for the improvements are well justified and the experiments are sound (extensive hyper-parameter grid search, high number of experiment repetitions, high number of benchmark functions). The key points of the proposed algorithm are the few number of hyper-parameters and its performance on multi-modal benchmark functions. Unfortunately, the proposed algorithm is not applied on a real-world problem and it is not compared to the Firefly Algorithm it is inspired by.

I suggest minor modifications to further improve the manuscript:

1) The proposed algorithm should be applied on a real-world problem. European Space Agency provides Global Trajectory Optimization Problems implemented in Matlab.

>> The proposed ERA is now applied on the Global Trajectory Optimization Problems.

2) A comparison with the Enhanced Firefly Algorithm [21] you are inspired by should be considered. The Matlab code for the Enhanced Firefly Algorithm is available on-line.

>> The proposed ERA is now compared with the enhanced firefly algorithm (FA) [21], which is called FA with courtship learning (FA-CL).

3) In Section 2 (line 138 page 6) you states that crossover favors exploitation while mutation favors exploration but in Section 1 (line 17 page 2) you states that crossover and mutation are responsible for exploration. Can you please be clearer?

>> Section 1 (line 17 page 2) is now revised: ~~The crossover and mutation are responsible for exploration, while elitism directs toward exploitation.~~ The mutation is responsible for exploration, while crossover and elitism direct toward exploitation.

4) Equation (7) in Section 2.3. (page 9) does not provide any information about the mutation operator employed.

>> Equation (7) is now equipped with the mutation operator provided in Equation (8).

5) The difference between multi-modal functions (ID=8 to 13) and fixed-dimension multi-modal functions (ID=14 to 23) is not clear. Is the difference only about decision space dimensions? In this case, it could be convenient to rename "fixed-dimension multi-modal functions" into "low-dimension multi-modal functions".

>> All the terms of ~~fixed-dimension multi-modal functions~~ are now revised into low-dimension multimodal functions.

6) In Section 3.1 it is stated that the results of 20 other benchmark functions are similar to those of functions 1, 8 and 22 (line 208, page 10). According to Table 2, it seems not to be the case. Indeed, for function ID=8 population size is 30 and proportion is 0.2 while for function ID=10 population size is 90 and proportion is 0.9.

>> Here, only three experimental results of the representative benchmark functions are shown and discussed: unimodal (Sphere, ID = 1), multimodal (Schwefel, ID = 8), and low-dimension multimodal (Shekel 7, ID = 22) ~~since the results of 20 other benchmark functions are similar to those three results~~ to see the behaviors of both parameters *p* and *s* in optimizing the three types of benchmark functions. The common parameter value of *p* is finally selected as a fixed-optimum value for all the benchmark functions. Meanwhile, the portion *s* is dynamically updated during the evolution process using a fitness-based adaptation scheme.

7) According to Figure 2, a low population size seems to be better to treat the 30-d Schwefel problem. According to Figure 3, a high population size seems to be better to treat the Shekel 4-d problem. I found counterintuitive that a lower population size is preferred to treat a more difficult problem (my first guess is a 30-d problem is more difficult than a 4-d problem). Could you please give a hypothesis to explain this observation?

>> Although its dimension is lower than the 30-D Schwefel, the 4-D Shekel has a broad flat area that makes some individuals in ERA may have the same fitness, which is hard to split them into the high- and the low-qualities (leading to a stagnation). Hence, a bigger population size is needed to escape from stagnation. The 2-D visualizations of the 23 classic benchmark functions is now provided in Figure 1 to give a better explanation for the hypothesis.

8) Typos and sentences that could be rephrased:

- They make GWO has a high exploitative searching strategy. (line 63, page 4)
- for each wolf to enhances the balance (line 70, page 4)
- an ability so solve (line 79, page 4)
- the best candidate as value of variable j (line 93, page 5)
- X_{m,LQ,i} and X_{m,HQ,n} is the (line 181, page 9)
- not all dimensions is used (line 183, page 9)
- a small portion [...] make (line 236, page 13)
- fixed-dimension unimodal functions (line 274, page 14)
- keeps evolves (line 341, page 21)
- to converge to the solution that very close to the Rao-1 (line 341, page 21)
- fixed-dimension unimodal (line 366, page 26)
- keeps evolving until finally converges (line 378, page 26)

>> The typos and sentences are now rephrased as follow:

- It has four phases, which are mathematically modeled into four behaviors: Harassing Prey, Hunting, Attacking, and Searching~~. They make GWO has a high exploitative searching strategy.~~ that create a high exploitative searching strategy.
- for each wolf to ~~enhances~~ enhance the balance (line 70, page 4)
- an ability ~~so~~ to solve (line 79, page 4)
- the best candidate as the value of variable j (line 93, page 5)
- X_{m,LQ,i} and X_{m,HQ,n} ~~is~~ are the (line 181, page 9)
- not all dimensions ~~is~~ are used (line 183, page 9)
- a small portion [...] ~~make~~ makes (line 236, page 13)
- ~~fixed-dimension~~ ~~unimodal~~ low-dimension multimodal functions (line 274, page 14)
- keeps ~~evolves~~ evolving (line 341, page 21)
- to converge to the solution that very ~~close~~ closes to the Rao-1 (line 341, page 21)
- ~~fixed-dimension unimodal~~ low-dimension multimodal (line 366, page 26)
- keeps evolving until finally ~~converges~~ converging (line 378, page 26)

Reviewer #2

The paper proposed an improved RAO algorithm called Evolutionary RAO Algorithm (ERA). ERA differs from RAO-1, RAO-2 and RAO-3 in that it has two additional schemes: (a) the first scheme is a population splitting mechanism which divides the main population into two sub-populations based on their fitness. (b) A crossover and mutation operators were also used in the proposed method. Moreover, a random walk was also used.

The performance of ERA has been validated by means of numerical experiments on well-known benchmark functions. Although the manuscript is well-organized and well-written, the following aspects of the paper needs major revisions:

Regarding the test functions:

The used test functions, indeed, are very well-known and widely used benchmark functions in the literature. However, the following points should be handled:

1) The authors should justify the choice of benchmark functions. Majority of the used test functions have symmetric search space boundaries, and in most of them the optimal point x* is in (0,0, …,0). Other test functions should be considered to investigate (a) the bias of the proposed algorithm toward the center of the search space, and (b) the effect of shift and rotation of the test function on the behavior of the proposed method. (c) the effect of noise in the fitness function.

>> The proposed ERA is now evaluated using the CEC-C06 2019 Benchmarks ‘‘The 100-Digit Challenge’’.

Regarding the simulation results and performance comparison:

2) It is not clear how the authors validated the difference between numerical results of counterpart algorithms. Authors are asked to conduct a non-parametric statistical analysis, i.e. Wilcoxon rank sum test, on the results and report the p-values.

>> Validations using Friedman mean rank (FRM) and Wilcoxon rank sum test (WRST) on the results are now provided and the *p*-values are now reported.

3) When reporting the results of the proposed algorithm, the authors conducted a considerable amount of experiments to find the sub-optimal values for p and s parameters. Although the proposed parameters are beneficial to the performance of the ERA, the used approach to find their best values for each benchmark function is not fair. Indeed, the process of finding the best values for the two parameters is a huge computational burden which simply neglected when judging about the superiority of the proposed method. Therefore, the authors should use p and s with the same values for all benchmark functions.

>> ERA and all the competitors are now evaluated using the same parameter setting of *p* and *s* for all the benchmark-functions to get fairness. Besides, they are compared on their best performances. Based on the previous research and a preliminary experiment performed in this research, the optimum values of *p* for Rao-1, Rao-2, and FA-CL are 20 while for Rao-3 and ERA are 40 and 60, respectively. A fitness-based adaptation scheme is now introduced in ERA to increase or decrease the portion *s* dynamically based on the best-so-far fitness during the evolution. If two consecutive best-so-far fitness shows an improvement, then the portion *s* is decreased to make ERA more exploitative. In contrast, if two consecutive best-so-far fitness shows a stagnation, then the portion *s* is increased to make ERA more explorative. Besides, both mutation-radius *a* and mutation-rate *b* are also dynamically updated using the same scheme. Hence, the Taguchi method is unnecessary to fine tune the parameters.

4) The computational complexity of the proposed ERA should be calculated and compared with those of the other algorithms.

>> The computational complexity of the proposed ERA is now provided in Subsection 2.6.

Reviewer #3

Manuscript No.: JOCSCI-D-21-00030

Manuscript title: Evolutionary Rao Algorithm

The above manuscript proposed Evolutionary Rao Algorithm and its application for solving unconstrained optimization problems. Several unconstrained benchmarks are considered. The manuscript lacks sufficient contribution and novelty and based on the following major comments, the manuscript is not ready for publication due the following comments:

1) The concept of sub-population will add another user parameter, and there will be question here, what is the optimal number of sub-population number? Regarding cross-over and mutation rates, both are considered as user parameters which are important factors. Therefore, user parameters are not considered as population and *s* which were fine tuned.

>> ERA is now equipped with a fitness-based adaptation scheme to increase or decrease the portion *s* dynamically based on the best-so-far fitness during the evolution. If two consecutive best-so-far fitness shows an improvement, then the portion *s* is decreased to make ERA more exploitative. In contrast, if two consecutive best-so-far fitness shows a stagnation, then the portion *s* is increased to make ERA more explorative. Besides, both mutation-radius *a* and mutation-rate *b* are also dynamically updated using the same scheme.

2) The benchmarks used are simple and very standard version of benchmarks. Those are not shifted, rotated and hybrid. They are in their simple format. It is strongly suggested to examine their improved method over CEC benchmark series.

>> The proposed ERA is now evaluated using the CEC-C06 2019 Benchmarks ‘‘The 100-Digit Challenge’’.

3) Talking about sensitivity analysis, the reviewer is not convinced with the methodology applied in the manuscript. It is mostly based on try and error various combinations of user parameters. It is strongly suggested to use well-known method such as Taguchi approach to fine tune initial parameters.

>> ERA is now run using the same *p* and *s* for all the benchmark functions. A fitness-based adaptation scheme is introduced to increase or decrease the portion *s* dynamically based on the best-so-far fitness during the evolution. If two consecutive best-so-far fitness shows an improvement, then the portion *s* is decreased to make ERA more exploitative. In contrast, if two consecutive best-so-far fitness shows a stagnation, then the portion *s* is increased to make ERA more explorative. Besides, both mutation-radius *a* and mutation-rate *b* are also dynamically updated using the same scheme. Hence, the Taguchi method is unnecessary to fine tune the parameters.

4) The manuscript only focuses on unconstrained benchmarks which is not acceptable at this moment. Ability of handling constrained benchmarks should be validated.

>> The ability of the proposed ERA in handling the constrained Global Trajectory Optimization Problems (GTOC) from European Space Agency is now validated.

5) Comparison pool is not sufficient and fair. It is intended to include more recent and competitive optimizers not only some variants of RAO. We wish to developed, improved an optimizer to make it better than the existing ones. This is the main purpose and target of improving/developing/hybridizing algorithms.

>> The proposed ERA is now compared with a recent optimizer called Firefly Algorithm with courtship learning (FA-CL) [21].

6) Concerning presenting simulation results, there are only statistical results such as mean, SD, worst, and best. There is no report of statistical tests such as T-test and Friedman. How can be sure about the significance of the obtained results?

>> Validations using Friedman mean rank (FRM) and Wilcoxon rank sum test (WRST) on the results are now provided and the *p*-values are now reported.

7) Future research is missing. Conclusions instead of conclusion.

>> The future research is now provided on two last sentences in the Conclusions: However, in the future, a new advanced adaptation scheme to update the population size dynamically throughout the evolutionary process as well as a better mutation scheme will be created to improve the performance of ERA. Besides, it will be comprehensively examined using more challenging benchmarks.
>> ~~Conclusion~~ Conclusions.

8) Please double proofread the entire manuscript. It can be seen some typo-mistakes and grammatical errors in the manuscript.

>> The entire manuscript is now double proofread. The typo-mistakes and grammatical errors are now corrected.

Highlights

- An evolutionary Rao algorithm (ERA) is proposed to enhance the three state-of-the-art metaheuristic Rao algorithms by introducing two new schemes
- The population is split into two subpopulations: high- and low-quality individuals to control searching strategy
- Two evolutionary operators: crossover and mutation operators are incorporated to give the exploitation and exploration strategies
- A fitness-based adaptation procedure is introduced to dynamically tune the three sensitive parameters to balance the exploitation and exploration
- Comprehensive examinations are performed using 38 benchmarks: 23 classic, 10 CEC-C06 2019, and 5 global trajectory optimization problems

Evolutionary Rao Algorithm

Suyanto Suyanto[a,*], Agung Toto Wibowo[a], Said Al Faraby[a], Siti Sa'adah[a], Rita Rismala[a]

[a]*School of Computing, Telkom University, Bandung, Indonesia*

Abstract

This paper proposes an evolutionary Rao algorithm (ERA) to enhance three state-of-the-art metaheuristic Rao algorithms (Rao-1, Rao-2, Rao-3) by introducing two new schemes. Firstly, the population is split into two sub-populations based on their qualities: high and low, with a particular portion. The high-quality sub-population searches for an optimum solution in an exploitative manner using a movement scheme used in the Rao-3 algorithm. Meanwhile, the low-quality one does in an explorative fashion using a new random walk. Secondly, two evolutionary operators: crossover and mutation, are incorporated to provide both exploitation and exploration strategies. A fitness-based adaptation is introduced to dynamically tune the three parameters: the portion of high-quality individuals, mutation radius, and mutation rate throughout the evolution, based on the improvement of best-so-far fitness. In contrast, the crossover is implemented using a standard random scheme. Comprehensive examinations using 38 benchmarks: twenty-three classic functions, ten CEC-C06 2019 benchmarks, and five global trajectory optimization problems show that the proposed ERA generally outperforms the four competitors: Rao-1, Rao-2, Rao-3, and firefly algorithm with courtship learning (FA-CL). Detailed inves-

This research is funded by the Directorate of Research and Community Service PPM, Telkom University, with grant number: 444/PNLT3/PPM/2020.

*Corresponding author

Email addresses: suyanto@telkomuniversity.ac.id (Suyanto Suyanto), agungtoto@telkomuniversity.ac.id (Agung Toto Wibowo), saidalfaraby@telkomuniversity.ac.id (Said Al Faraby), sitisaadah@telkomuniversity.ac.id (Siti Sa'adah), ritaris@telkomuniversity.ac.id (Rita Rismala)

tigations indicate that both proposed schemes work very well to make ERA evolves in an exploitative manner, which is created by a high portion of high-quality individuals and the crossover operator, and avoids being trapped on the local optimum solutions in an explorative manner, which is generated by a high portion of low-quality individuals and the mutation operator. Finally, the adaptation scheme effectively controls the exploitation-exploration balance by dynamically tuning the portion, mutation radius, and mutation rate throughout the evolution process.

Keywords: evolutionary Rao algorithm, exploitation-exploration balance, fitness-based adaptation scheme, random walk, two subpopulations

---

1. Introduction

The metaheuristic optimization algorithms that can be categorized into two groups: evolutionary algorithms (EAs) and swarm intelligence (SI) algorithms [1]. EAs are inspired by both evolution and natural selection, such as Genetic Algorithm (GA) [2], [3], Evolution Strategies (ES) [4], [5], and Differential Evolution (DE) [6]. Meanwhile, SI algorithms are inspired by a natural swarm, such as Particle Swarm Optimization (PSO) [7], [8], Firefly Algorithm (FA) [9], [10], Grey Wolf Optimizer (GWO) [11], [12], and Ant Lion Optimization (ALO) [13].

GA is one of the most popular EAs introduced in the 1970s [14]. It uses both evolution and natural selection that are applied to its population over generations. A population consists of some individual chromosomes, each representing a candidate solution. The new chromosomes in a generation are either some of the best chromosomes (elitism) in the previous generation or generated by genetic operations, such as crossover and mutation. The crossover takes two chromosomes and produces one offspring inherited part of chromosome values from each parent. In contrast, the mutation is randomly changing some values in a chromosome. The mutation is responsible for exploration, while crossover and elitism direct toward exploitation. GA can avoid being trapped in the local optima. It is also applicable to non-differentiable and high dimensionality functions. On the other hand, it converges slowly because of the highly-random operations that do not give a clear direction to find the global optimum solution quickly. However, various improvement schemes have been proposed to overcome the drawback, such as a concept of human-like constrained-mating [15] that creates a more explorative search strategy.

In 1995, the Particle Swarm Optimisation (PSO) was introduced by Kennedy and Elberhart [16]. The movements of the particles in searching for a global optimum mimics the behavior of bird flocking and fish schooling. PSO is one of the most popular SI algorithms since it has three advantages: easy to implement, few parameters that are simply tuned, and effective in searching the global optimum solution since it has a clearer direction than GA. However, it tends to prematurely converge on a local optimum in optimizing a multimodal function since it uses a static finite leader and group based on a linear movement. Therefore, some strategies are developed to tackle the issue, such as a learning structure [17] to decouple exploration and exploitation and a dynamic updating of the inertia weights [18] to control the convergence.

In 2009, the Firefly Algorithm (FA) was proposed [19]. In FA, each firefly will be attracted to all other brighter (better) fireflies, not only to the global best like in PSO. Also, the brighter firefly's attractiveness is decreased proportioned to the distance between the two fireflies due to the light absorption. Since the fireflies will usually be attracted more to their brighter neighbor than the further away brightest individual, the exploration is more effective than PSO. In other words, FA uses a dynamic leader and group based on a nonlinear movement. Moreover, FA can be turned into PSO by setting the light absorption parameter such that every firefly can be seen clearly by all other fireflies. Consequently, all fireflies will be attracted to the brightest one (global best). In some experiments, FA shows better performance than PSO due to two critical characteristics [20]: 1) FA usually divides its population into a subgroup, 2) By not having an explicit global best, FA can avoid premature convergence. Several improved schemes are created to enhance the FA performance, such as a courtship learning framework [21], where the population is divided into subpopulations: female and male, to improve the convergence speed and solution accuracy. Another improvement scheme is the best neighbor guided strategy [22], where each firefly is attracted to the best firefly among some randomly chosen neighbors to decrease the firefly oscillations in every attraction-induced migration stage as well as increase the probability of the guidance a new better direction.

In 2014, Grey Wolf Optimization (GWO) was introduced by Mirjalili [23]. It is inspired by both the social hierarchy and hunting methods of grey wolves (GWs). The hierarchy of GWs has four groups: alpha, beta, delta, and omegas. GWO selects the three fittest wolves (best solutions) as the alpha, beta, and delta, while the rest as omegas. The hunting process of GWs is guided by the three fittest wolves. All omegas follow them. It has four phases, which are mathematically modeled into four behaviors: Harassing Prey, Hunting, Attacking, and Searching, that create a high exploitative searching strategy. It quickly converges to an optimum solution for unimodal functions. However, it suffers from multimodal functions since it has a low explorative movement. Therefore, some variants of GWO are developed by incorporating various mechanisms/operators, such as differential evolution with elimination mechanism [24], simulated annealing [25], or refraction learning operator [26]. GWO can also be improved using a dimension learning-based hunting movement strategy [27], which uses a different approach to construct a neighborhood for each wolf to enhance the balance of local and global searches and maintain diversity.

In 2015, Ant Lion Optimizer (ALO) was proposed by Mirjalili [28]. ALO mimics the interaction between antlions and ants in the trap, where ants move over the search space and antlions hunt them and become fitter using traps. A new random walk is introduced to model the ant's movement as they move stochastically in nature to find some food. It has high exploitation and convergence speed because of the adaptive boundary shrinking mechanism and elitism. It also high exploration due to the random walk and roulette wheel selection mechanisms. However, although it has few parameters, some schemes and movements make ALO seems too-complicated. Hence, some versions of ALO are created by modifying, hybridizing, and providing an ability to solve a multi-objective problem [13].

In 2020, the metaphor-less optimization methods called Rao algorithms were proposed by Ravipudi Venkata Rao [29]. The Rao Algorithms use both best and worst solutions in each iteration and the random interactions among the candidate solutions to quickly find an optimum solution. They need two standard parameters: population size and a maximum number of evaluations that easy to adjust. They drop many parameters used in the previous metaphor-based algorithms, such as cohesion, intensity, probability, and other commonly challenging parameters to tune carefully.

The Rao algorithms have three variants: Rao-1, Rao-2, and Rao-3, which respectively use three different equations below:

$$X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,best,i} - X_{j,worst,i}) \tag{1}$$

$$\begin{aligned} X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,best,i} - X_{j,worst,i}) + \\ r_{2,j,i}(|X_{j,k,i} \text{ or } X_{j,l,i}| - |X_{j,l,i} \text{ or } X_{j,k,i}|), \end{aligned} \tag{2}$$

$$\begin{aligned} X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,best,i} - |X_{j,worst,i}|) + \\ r_{2,j,i}(|X_{j,k,i} \text{ or } X_{j,l,i}| - (X_{j,l,i} \text{ or } X_{j,k,i})), \end{aligned} \tag{3}$$

where $X_{j,best,i}$ represents the best candidate as the value of variable j, and $X_{j,worst,i}$ represents the worst candidate as value of variable j, both throughout the i-th iteration. $X'_{j,k,i}$ is the updated value after the equation, and both $r_{1,j,i}$ as well as $r_{2,j,i}$ are randomly generated in [0,1] for the j-th variable throughout the i-th iteration. In the term $|X_{j,k,i}$ or $X_{j,l,i}|$, the candidate solution k is compared to another candidate l, which is randomly selected from the available candidates in the population. The term $|X_{j,k,i}|$ is selected if k is fitter than l. Otherwise, the $|X_{j,l,i}|$ is chosen. The same rule is applied to the second the term ($X_{j,l,i}$ or $X_{j,k,i}$).

All formulas used in the three Rao algorithms are similar to GWO, making them more exploitative than explorative. Using both best and worst solutions, they converge to an optimum solution for unimodal functions more quickly than GWO. However, with low explorative movement, they can be worse for multi-modal functions. As described in [29], Rao is easy to get stuck in multimodal functions. Rao-3 gives a better solution in the Schwefel function from the six benchmark multimodal-functions and much worse for the other five benchmark multimodal-functions.

Therefore, in this research, an evolutionary Rao algorithm (ERA) is proposed to enhance the three original Rao algorithms by introducing two additional schemes. Firstly, the population is split into two sub-populations based on their qualities: high and low, with a particular portion depending on the given problem. The high-quality sub-population searches for an optimum solution in an exploitative manner using a movement scheme used in the Rao-3 algorithm. Meanwhile, the low-quality one does in an explorative fashion using a new random walk introduced in this research. This scheme is similar to the courtship learning framework in the Enhanced FA [21], where the population is also divided into two subpopulations: female and male, but ERA uses a pre-defined specific portion. Secondly, two evolutionary operators: crossover and mutation, are used to give exploitation and exploration searching strategies. A fitness-based adaptation is introduced to dynamically tune the the portion of high-quality individuals, mutation radius, and mutation rate during the evolution. Meanwhile, the crossover is implemented using a random scheme with the common probabilistic values that do not create any additional parameters. The ERA is finally examined and compared to the three original Rao algorithms [29] as well as the firefly algorithm with courtship learning (FA-CL) [21] using three groups of benchmark functions: 1) the classic benchmark functions that contain seven unimodal, six multimodal, and ten low-dimension multimodal; 2) the CEC-C06 2019 test suites that consists of ten benchmark functions [30]; and 3) the global trajectory optimization problems provided by European Space Agency that contains five real problems of Cassini1, GTOC1, Messenger, Sagas, and Cassini2 [31].

2. Proposed Evolutionary Rao Algorithm

The pseudo-code of ERA is illustrated in Algorithm 1. In the initial phase, define the fixed population size p, the initial portion of high-quality (HQ) individuals $s = 0.5$, the initial mutation radius $a = 0.5$, the initial mutation rate $b = 0.9$, and randomly initialize the population of p individuals. In the next phase, the evolution is performed until a stopping condition is reached, such as when the number of evaluations is equal to the given maximum limit.

In each generation, six steps are carried out. Firstly, the quality of each individual is calculated; and their quality-ranks are then sorted in the descending mode. Secondly, the population is split into two sub-populations: high-quality (HQ) and low-quality (LQ), with the defined portion s, and both the best individual X_{best} and the worst individual X_{worst} are selected. Thirdly, each HQ individual is moved to follow the X_{best} using Eq. (3). Fourthly, the fittest HQ individual is selected as the BestHQ, and then one of the two evolutionary operators is chosen: crossover (exploitative) or mutation (explorative), to move the X_{best}. Fifthly, each LQ individual is moved using a new random walk. Finally, the fitness-based adaptation is performed by updating s, a, and b based on the improvement or stagnation of two consecutive best-so-far fitness.

2.1. Two sub-populations

The population of p candidate solutions (individuals) is split into two sub-populations based on their qualities: high and low, with a proper portion based on the given problem. The high-quality (HQ) sub-population searches for an optimum solution in an exploitative manner using the same movement scheme as in the Rao-3 algorithm. Meanwhile, the low-quality (LQ) one does in an explorative fashion using a new random walk introduced in this research. Hence, this scheme creates a new parameter s: the portion of high and low-quality individuals in the population. It is in the interval (0, 1) and easy to adjust. Hypothetically, it should be high (more than 0.5) to make ERA more exploitative and faster to optimize the unimodal functions. In contrast, it must be low (less

Algorithm 1: Evolutionary Rao Algorithm

Result: X_{best} as the optimum solution

Set p as the fixed population size (number of individuals);

Set $s = 0.5$, $a = 0.5$, and $b = 0.9$ as the initial values of high-quality (HQ) individuals portion, mutation radius, and mutation rate, respectively;

Randomly initialize the population of p individuals;

```
while StoppingCondition = false do
    for each individual, calculate its quality and then sort the
      quality-ranks in the descending mode;
    Select the fittest individual as the X_best;
    Select the most fit individuals with the defined portion s as the HQ
      and the rests as the low-quality (LQ) individuals;
    Select the lowest-quality individual as the X_worst;
    for each HQ individual, move it to follow the X_best using Eq. 3;
    Select the fittest HQ individual as the BestHQ;
    if rand > 0.5 then
        Offsprings = Crossover(BestHQ, X_best);
        Replacement(BestHQ, X_best, Offsprings);
    else
        Offspring = Mutation(X_best);
        Replacement(X_best, Offspring);
    end
    for each LQ individual move it to follow or distract a randomly
      selected HQ individual on the half of dimensions using Eq. (9);
    if two consecutive best-so-far fitness show an improvement then
        Increase s, but decrease a and b, using Eq. (10), (11), and (12);
    else
        Decrease s, but increase a and b, using Eq. (10), (11), and (12);
        Mutate (1 − s) × p low-quality individuals;
    end
end
```

than 0.5) to make ERA more explorative to solve the multimodal functions. A fitness-based adaptation scheme is proposed to increase or decrease the portion s automatically based on the best-so-far fitness during the evolution. If two consecutive best-so-far fitness values show an improvement, then the portion s is decreased to make ERA more exploitative. In contrast, if two consecutive best-so-far fitness shows a stagnation, then the portion s is increased to make ERA more explorative. A detailed explanation will be provided in Subsection 2.5.

Furthermore, the population of p individuals is split into two subpopulations: the high-quality subpopulation of h individuals and the low-quality subpopulation of l individuals, which are calculated as

$$h = \lfloor (p-1) \times s \rfloor, \tag{4}$$

$$l = (p-1) - h, \tag{5}$$

where s is the portion of HQ individuals in the population. However, both Eq. (4) and Eq. (5) may produce zero for either h or l if the portion s is too-small or too-high. Hence, an enforcement procedure is implemented to ensure that a too-small s makes the HQ sub-population consists of at least two individuals, and a too-big s also makes the LQ sub-population contains at least two individuals.

2.2. Crossover

The crossover is implemented using a whole arithmetic crossover, which is defined as

$$\begin{aligned} X' &= r \cdot X + (1-r) \cdot Y \\ Y' &= r \cdot Y + (1-r) \cdot X \end{aligned} \tag{6}$$

where r is a randomly generated number in the interval (0, 1), which should be not equal to 0.5 to prevent generating the same two offsprings (new individuals); if $r = 0.5$, then both offsprings X' and Y' are the same as the average of both current individuals X and Y. Hence, this crossover scheme does not need any user parameter.

2.3. Mutation

The mutation is simply implemented using a creep mutation by adding a small value (positive or negative) to each mutated element. The small value is randomly generated using a Gaussian probability that is symmetric, distributed on 0, and has a high probability for the smaller values. The creep mutation is defined as

$$\langle x_1, x_2, ...x_n \rangle \rightarrow \langle x'_1, x'_2, ...x'_n \rangle, \tag{7}$$

$$x'_i = \begin{cases} x_i + (2r_1 - 1) \times a|U_i - L_i|, & \text{if } r_2 < b \\ x_i, & \text{otherwise,} \end{cases} \tag{8}$$

where $x_1, x_2, ...x_n \in [L_i, U_i]$, L_i and U_i are the lower and upper bounds of the interval of the ith element (variable or dimension), r_1 and r_2 are random values with the normal distribution in the interval [0, 1], and a and b are the mutation radius and the mutation rate, respectively, which are automatically tuned using a fitness-based adaptation scheme that will be described in Subsection 2.5.

2.4. Random walk

To provide an ability to search for an optimum solution in an explorative manner, each LQ individual is moved using a new random walk formulated as

$$X'_{m,LQ,i} = X_{m,LQ,i} + r_{1,m,i}(X_{m,HQ,n} - X_{m,LQ,i}) \tag{9}$$

where $X_{m,LQ,i}$ and $X_{m,HQ,n}$ is the LQ individual i and the HQ individual n (randomly selected from the high-quality sub-population), respectively, and m is the randomly selected dimension; not all dimensions are used here to make this random walk more explorative.

2.5. Fitness-based adaptation scheme

Based on the above description, ERA has four parameters: population size p, portion s, mutation-radius a, and mutation-rate b. Hypothetically, p is the most robust parameter. In contrast, s, a, and b are estimated quite sensitive since they control the exploration strategy. Therefore, these three parameters are designed to be tuned adaptively during the evolution. A new simple fitness-based adaptation scheme based on the fitness values of the best-so-far individual is proposed for this purpose. If two consecutive best-so-far fitness values show an improvement, then s is increased, but both a and b are decreased, to make ERA more exploitative. In contrast, if two consecutive best-so-far fitness shows a stagnation, then s is decreased, both a and b are increased to make ERA more explorative, and all low-quality individuals are mutated using both new a and b to spread them in new locations. The increment and decrement are formulated as follow:

$$s' = \begin{cases} s \times (1 - \frac{\triangle f_1 + \triangle f_2}{2}), \text{ if } \triangle f_1 > 0 \text{ and } \triangle f_2 > 0 \\ s \times 0.97, \text{ if } \triangle f_1 = 0 \text{ and } \triangle f_2 = 0 \end{cases} \tag{10}$$

$$a' = \begin{cases} a \times 0.97, \text{ if } \triangle f_1 > 0 \text{ and } \triangle f_2 > 0 \\ a \times 1.03, \text{ if } \triangle f_1 = 0 \text{ and } \triangle f_2 = 0 \end{cases} \tag{11}$$

$$b' = \begin{cases} b \times 0.97, \text{ if } \triangle f_1 > 0 \text{ and } \triangle f_2 > 0 \\ b \times 1.03, \text{ if } \triangle f_1 = 0 \text{ and } \triangle f_2 = 0 \end{cases} \tag{12}$$

where $\triangle f_1 = \frac{|f_1 - f_2|}{f_1}$ and $\triangle f_2 = \frac{|f_2 - f_3|}{f_2}$ are the first and the second differences of the fitness values of two consecutive generations during the evolution process, respectively.

Moreover, the initial, minimum, and maximum values for those three parameters can be easily defined. Since the characteristics of the given problem are unknown, then the initial portion s is set as 0.5, while the minimum and the maximum values are set to 0.1 and 0.9, respectively. Next, both minimum and maximum values of a are set as 0.05 and 0.5, respectively. It means the mutation of an element (dimension) can occur in the radius of 5% to 50% out of the search space. In other words, an individual can be mutated at the maximum range of [-0.5, 0.5] in the search space. Hence, the mutation can cover the whole search space. Next, the initial value of a is tuned as 0.5 to provide the maximum exploration in the beginning iterations of the evolution process. Finally, b is defined in the interval [0.1, 0.9], and its initial value is 0.9 to maximize the exploration strategy in the beginning evolution process. Using the maximum mutation radius and rate, ERA can have a high-exploration ability to handle the effects of shift and rotation of the test functions, such as in the CEC-C06 2019 benchmark functions.

2.6. Complexity analysis of ERA

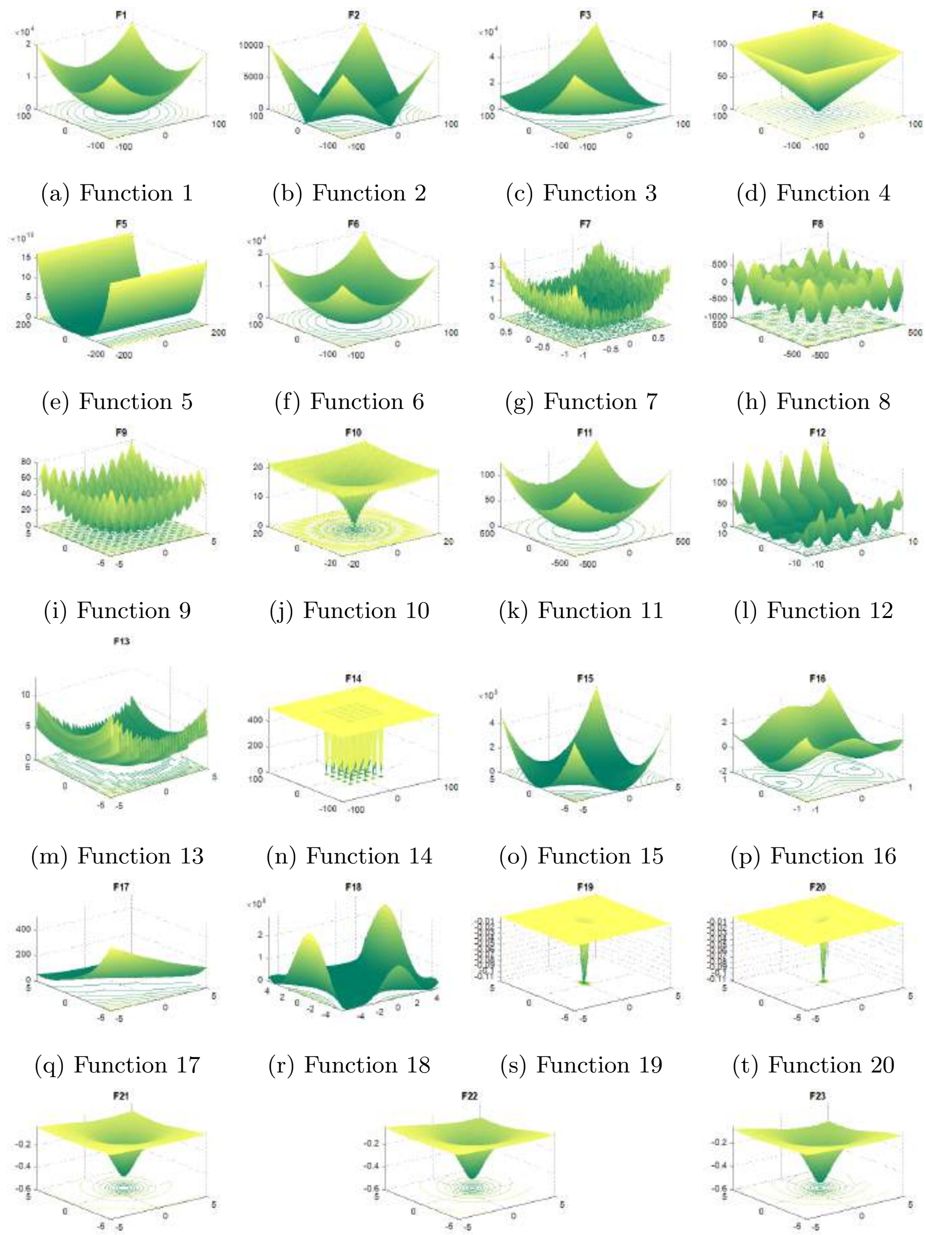
The mathematical complexity of ERA can be analyzed as follows. For each iteration, ERA has a time complexity of $O(p \times n + p \times c + \log p)$, where p is the population size, n is the dimension of the given problem, c is the complexity of the objective function calculation, and $\log p$ is the complexity of the fitness sorting to split the population into HQ and LQ sub-populations. It is clear that compared to the original Rao, ERA is slightly more complicated because of the additional sorting complexity of $\log p$. Meanwhile, the complexity of the fitness-based adaptation scheme can be ignored since it is quite low; it only contains addition, substraction, and logical operations.

3. Results and Discussion

In this research, twenty-three benchmark functions: seven unimodal, six multimodal, and ten low-dimension multimodal functions [29] are used to investigate both exploitation and exploration abilities of the proposed ERA. Table 1 illustrates the benchmark functions with their identities (ID), names, types, dimensions, ranges, and global optimum values f_{min}. Meanwhile, their two-dimensional views are illustrated in Figure 1. Seven benchmark functions, with ID = 1 to 7, are unimodal to examine the exploitation ability. Next, six benchmark functions, ID = 8 to 13, are multimodal, with many local optima increasing as the dimension increases, to evaluate the exploration ability. Finally, ten functions, ID = 14 to 23, are low-dimension multimodal (LDM) to investigate the exploration ability in the case of low-dimension optimization problems.

Table 1: Twenty three classic benchmark functions

Func	Name	Type	Dim	Range	f_{min}
CF1	Sphere	Unimodal	30	[-100, 100]	0
CF2	Schwefel 2.22	Unimodal	30	[-100, 100]	0
CF3	Schwefel 1.2	Unimodal	30	[-100, 100]	0
CF4	Schwefel 2.21	Unimodal	30	[-100, 100]	0
CF5	Rosenbrock	Unimodal	30	[-30, 30]	0
CF6	Step	Unimodal	30	[-100, 100]	0
CF7	Quartic	Unimodal	30	[-1.28, 1.28]	0
CF8	Schwefel	Multimodal	30	[-500, 500]	-418.9829 $\times$ Dim
CF9	Rastrigin	Multimodal	30	[-5.12, 5.12]	0
CF10	Ackley	Multimodal	30	[-32, 32]	0
CF11	Griewank	Multimodal	30	[-600, 600]	0
CF12	Penalized	Multimodal	30	[-50, 50]	0
CF13	Penalized2	Multimodal	30	[-50, 50]	0
CF14	Foxholes	LDM	2	[-65, 65]	0.998
CF15	Kowalik	LDM	4	[-5, 5]	0.0003
CF16	Six Hump Camel	LDM	2	[-5, 5]	-1.0316
CF17	Branin	LDM	2	[-5, 5]	0.398
CF18	GoldStein-Price	LDM	2	[-2, 2]	3
CF19	Hartman 3	LDM	3	[0, 1]	-3.86
CF20	Hartman 6	LDM	6	[0, 1]	-3.32
CF21	Shekel 5	LDM	4	[0, 10]	-10.1532
CF22	Shekel 7	LDM	4	[0, 10]	-10.4029
CF23	Shekel 10	LDM	4	[0, 10]	-10.5364



(a) Function 1 (b) Function 2 (c) Function 3 (d) Function 4

(e) Function 5 (f) Function 6 (g) Function 7 (h) Function 8

(i) Function 9 (j) Function 10 (k) Function 11 (l) Function 12

(m) Function 13 (n) Function 14 (o) Function 15 (p) Function 16

(q) Function 17 (r) Function 18 (s) Function 19 (t) Function 20

(u) Function 21 (v) Function 22 (w) Function 23

Figure 1: Twenty three classic benchmark functions CF1 to CF23

3.1. Preliminary observations

First, two parameters of ERA: population p and portion s, are observed to see their behaviors in optimizing the twenty-three classic benchmark functions. For each function, ninety experiments are performed using combination of ten values of p = 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 and nine values of s = 0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 0.8, 0.9, which can be defined as pairs of (10, 0.1), (10, 0.2), ..., (100, 0.9). For each experiment, the maximum number of function evaluations is set to 30,000 with ten runs to reduce the coincidence. Here, only three experimental results of the representative benchmark functions are shown and discussed, namely unimodal (Sphere, ID = 1), multimodal (Schwefel, ID = 8), and low-dimension multimodal (Shekel 7, ID = 22), to see the behaviors of both parameters p and s in optimizing those three types of benchmark functions. The common parameter value of p is finally selected as a fixed-optimum value for all the benchmark functions. Meanwhile, the portion s is dynamically updated during the evolution process using a fitness-based adaptation scheme.

Figure 2 illustrates the experimental results for the problem of searching a minimum solution to a unimodal function of Sphere (ID = 1), where the vertical axis uses log(mean solution) to ensure the bar chart clearly shows all results from the ninety experiments. It can be seen that a too-small (10) or a big population p (30 to 100) makes the ERA produces a bad solution. The bigger the p, the worse the solution. A small portion s (0.5 or less) also yields a poor solution. The smaller the s, the worse the solution. Hence, the combination of a too-big p and a too-small s is not recommended. The optimum combination is reached on p = 20 and s = 0.8. This result proves that a big portion of high-quality individuals in the small population makes the proposed ERA more exploitative and faster to find the optimum solution.

Next, Figure 3 illustrates the ninety experimental results for the problem of minimizing a multimodal function of Schwefel (ID = 8). It informs that the portion s is sensitive, but the population size p is not; the bigger the s, the worse the solution. A too-big portion s drastically reduces the solution quality. The optimum combination is reached on $p = 30$ and $s = 0.2$. This result proves

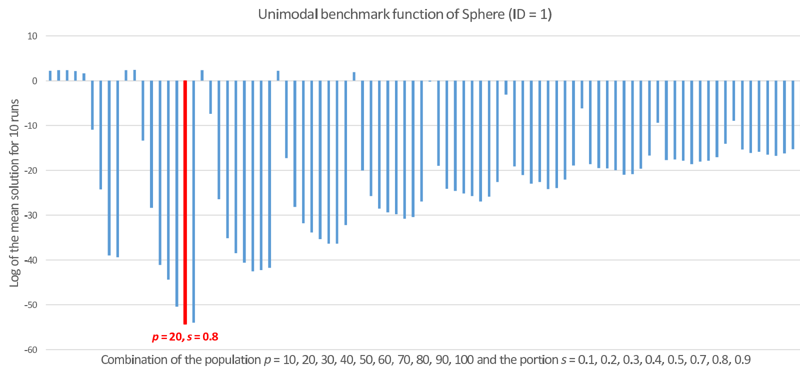


Figure 2: Parameter tuning for a unimodal benchmark function of Sphere (ID = 1)

that a small portion of high-quality individuals in the small population makes the proposed ERA more explorative and faster to find the optimum solution to the multimodal functions with many local optima.

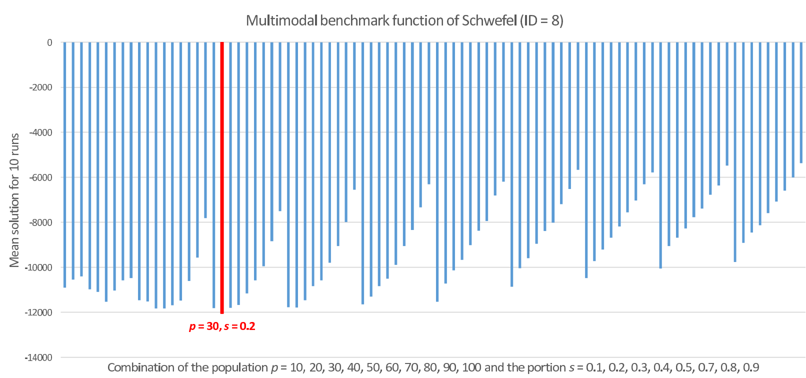


Figure 3: Parameter tuning for a multimodal benchmark function of Schwefel (ID = 8)

Finally, Figure 4 illustrates the ninety experimental results for the problem of minimizing a low-dimension multimodal function of Shekel 7 (ID = 22). It also informs that the portion s is sensitive, but the population size p is not; the bigger the s, the worse the solution. A too-big portion s drastically reduces the solution quality. The optimum combination is reached on a big $p = 100$ and a low $s = 0.2$. However, a smaller p up to 20 or 30 also gives a good solution. This result informs that a small portion of high-quality individuals in the big population makes ERA more explorative. Hence, it can search for an optimum solution to the low-dimension multimodal functions with a wide flat area.

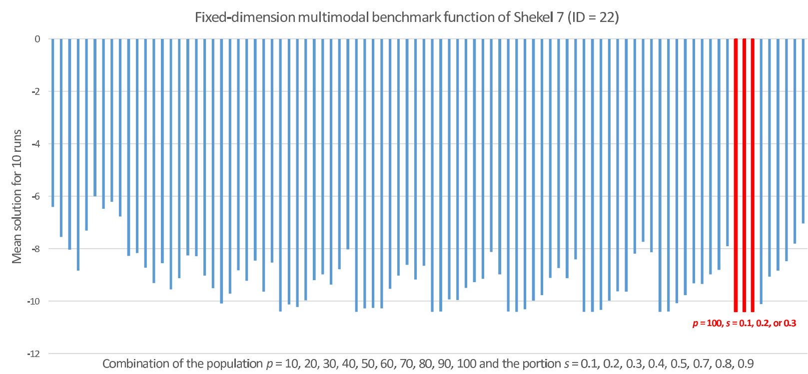


Figure 4: Parameter tuning for a low-dimension multimodal benchmark function of Shekel 7 (ID = 22)

The three observations above prove the hypothesis that p is more robust than s. Therefore, the adaptation scheme is applied on s instead of p. A fitness-based adaptation of population size introduced in [32] is reported can improve the performance of the differential evolution, but that scheme is not used here since it will increase the complexity of ERA. Thus, p is designed to be a fixed value and tuned manually by doing a few experiments.

3.2. Parameter settings

Based on the research in [21], the best population size for FA-CL is 20. Thus, the parameter setting is focused on Rao-1, Rao-2, Rao-3, and ERA. Here, ten experiments with $p = 10, 20, ..., 100$ are carried out to find the optimum p for each algorithm based on the Friedman Mean Rank (FMR).

Figure 5 illustrates the experimental results. The behavior of p is similar for Rao-1 and Rao-2. The smaller the p, the better the rank. The optimum value is reached on $p = 20$ for both algorithms. Meanwhile, p gives a different effect for Rao-3 that achieves the optimum value on $p = 40$. It also shows the different impacts for ERA, which gets the optimum value on $p = 60$. Finally, the parameter settings for ERA and the other algorithms are listed in Table 2.

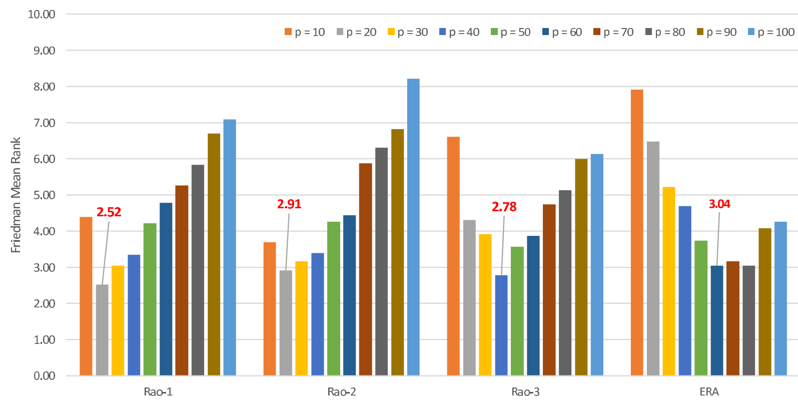


Figure 5: Friedman mean rank calculated using ten different population sizes p for each algorithm

Table 2: Parameter settings

Algorithm	Parameter settings
Rao-1	$p = 20$
Rao-2	$p = 20$
Rao-3	$p = 40$
FA-CL	$p = 20, \alpha = 0.5, \beta_{min} = 0.2, \beta = 1, \gamma = 1$
ERA	$p = 60, s = 0.5, a = 0.5, b = 0.9$

3.3. Evaluation on classic benchmark functions

First, the proposed ERA is then examined and compared with four other algorithms: Rao-1, Rao-2, Rao-3, and FA-CL to search the minimum solutions to the twenty-three benchmark functions listed in Table 1. For each benchmark function, the maximum number of function evaluations is set to 30,000 with 30 runs to reduce the coincidence. The random seeds of the 30 initial populations (for each benchmark function) are the same when the algorithms use the same population size p to get fairness. Otherwise, they are different. The Matlab source-codes used in the Rao-1, Rao-2 and Rao-3 refer to [29] while the one used in FA-CL refers to [21]. Meanwhile, the optimum parameter settings for all algorithms are described in Subsection 3.2. Table 3 illustrates the examination results based on five metrics (Met): best solution, worst solution, mean solution, standard deviation (STD), and mean function evaluations (MFE).

Based on the two metrics, mean solution and STD, for the seven unimodal functions, ID = 1 to 7, the proposed ERA commonly outperforms all the other algorithms for the four functions with ID = 4, 5, 6, and 7. Unfortunately, it is worse than Rao-3 and Rao-1 for two functions with ID = 1 and 2. Besides, it is much worse than Rao-1 for the function ID = 3.

Next, the investigation on the six multimodal functions, ID = 8 to 13, informs that the proposed ERA also generally outperforms the competitors, where it achieves much lower mean solutions for three functions with ID = 9, 10, and 12. It is slightly worse than Rao-3 and Rao-2 for the function ID = 8. It is much worse than Rao-1 and Rao-3 for the function ID = 11 and 13, respectively.

Finally, the investigation on the ten low-dimension multimodal functions, ID = 14 to 23, shows that the proposed ERA mostly gives better or equal mean solutions than the competitors. It reaches the best solutions for the three benchmark functions with ID = 20, 21 and 23. It gives the same or similar global solutions, with quite low MFE, as the three Rao algorithms for the benchmark function with ID = 16, 17, 18, and 19. It is slightly worse than Rao-1 or Rao-3 only for three benchmark functions (ID = 14, 15, and 22).

Table 3 Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for 23 classic benchmark functions

ID	**Metric**	**Rao-1**	**Rao-2**	**Rao-3**	**FA-CL**	**ERA**
1	Best	1.44026E-13	0.000206616	1.37198E-16	3360.332345	1.44407E-11
	Worst	1.56215E-11	0.05322429	3.57068E-13	7391.524099	7.07827E-10
	Mean	1.6427E-12	0.007910073	2.93612E-14	5494.642553	2.03341E-10
	STD	2.94821E-12	0.012500245	6.93888E-14	897.6329345	1.97333E-10
	MFE	30000	30000	30000	30176.8	30032

Table 3 Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for 23 classic benchmark functions

ID	Metric	Rao-1	Rao-2	Rao-3	FA-CL	ERA
2	Best	9.88458E-08	0.046106637	3.48002E-09	20.78017429	2.1222E-06
	Worst	2.84524E-05	81.77760505	1.96246E-07	36.38814153	2.38276E-05
	Mean	1.75865E-06	6.122312306	4.70062E-08	31.53033206	8.63966E-06
	STD	5.11921E-06	16.17762336	5.36523E-08	3.444133931	4.73519E-06
	MFE	30000	30000	30000	30182.46667	30028.2
3	Best	29.58384537	24308.67108	5351.839864	6485.004266	983.5012373
	Worst	410.0667787	45693.37909	19092.21252	14405.71624	3470.902824
	Mean	149.4341504	35855.17634	11301.74245	10486.02575	2303.225697
	STD	105.4899095	5718.824871	3745.250011	1944.35227	728.8781476
	MFE	30000	30000	30000	30208.9	30023.6
4	Best	1.201443048	7.137837854	0.042098173	21.36789617	0.042019247
	Worst	12.36146637	28.08331631	59.65566877	36.06694776	0.300012918
	Mean	5.214910493	16.15411285	7.347592214	28.9938981	0.124708246
	STD	3.489736613	4.841469952	13.05854573	3.59291026	0.061288909
	MFE	30000	30000	30000	30226.43333	30025.6
5	Best	0.287292008	0.439075353	12.58554648	226704.144	18.30326575
	Worst	93.46438644	3019.406575	100.1684583	2564425.907	109.1279756
	Mean	35.67946414	130.9427468	35.87671438	1261919.835	31.40688996
	STD	29.6735389	546.4756884	27.26289807	584515.0346	17.69375467
	MFE	30000	30000	30000	30203.33333	30029.4
6	Best	2	0	0	3792	0
	Worst	53	12	3	7639	2
	Mean	10.2	1.833333333	0.3	5555.266667	0.2
	STD	9.219170432	3.006697505	0.651258728	1099.28108	0.484234198
	MFE	30000	28073.33333	13776	30144.96667	12368
7	Best	0.03452811	0.03404685	0.005036199	1.530176956	0.004682426
	Worst	0.211389877	0.168091645	0.081089247	4.106762973	0.036293801
	Mean	0.080890625	0.093085738	0.019933199	2.624059197	0.013211183
	STD	0.036979081	0.031284187	0.016182882	0.659081008	0.007750259
	MFE	30000	30000	30000	30145.5	30027.2
8	Best	-10682.09946	-10239.35633	-11345.58004	-4377.087113	-8879.935043
	Worst	-3893.432932	-5125.91502	-3869.781006	-3291.297315	-6882.950077
	Mean	-6470.266849	-8027.033295	-8325.725086	-3701.706264	-7753.907755
	STD	2090.801691	1321.261016	2361.956999	288.5432924	413.9236488
	MFE	30000	30000	30000	30163.16667	30038.8
9	Best	82.58144051	183.9047143	174.6873065	189.5462566	11.82239455

Table 3 Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for 23 classic benchmark functions

ID	Metric	Rao-1	Rao-2	Rao-3	FA-CL	ERA
	Worst	275.100287	283.1739045	249.2618453	242.3410246	44.09695302
	Mean	211.4806106	238.8109701	203.3125985	216.0038257	29.26076862
	STD	41.58262643	25.05505101	17.48088927	12.57640716	7.210275076
	MFE	30000	30000	30000	30246.93333	30033.6
10	Best	1.340421288	0.01602575	4.51465E-09	11.80926578	9.14087E-07
	Worst	19.96317829	19.96048248	0.931304602	13.48685262	1.87018E-05
	Mean	3.544848527	6.029369132	0.062087072	12.71855001	4.61579E-06
	STD	4.523133052	8.827128996	0.236279524	0.503095444	3.42908E-06
	MFE	30000	30000	30000	30174.03333	30034.8
11	Best	3.87024E-13	0.000684082	4.67404E-14	18.67629168	5.55651E-11
	Worst	0.070984139	0.741672368	0.569327929	68.09827573	0.343918782
	Mean	0.016380538	0.480608522	0.125402819	41.81819117	0.093842602
	STD	0.016674644	0.220909193	0.12739394	11.89438954	0.089296594
	MFE	30000	30000	30000	30176.23333	30034.8
12	Best	2.95944E-12	0.101041766	0.320579961	1112.053147	0.031510408
	Worst	25.77634972	15.3687038	2.587976377	219089.505	2.00220998
	Mean	3.326291084	5.096597329	0.818413527	48693.6658	0.37438139
	STD	5.791910477	3.953017932	0.57337885	53495.37265	0.390735094
	MFE	30000	30000	30000	30234.96667	30038.6
13	Best	1.46599E-12	4.87385E-12	2.56084E-17	75765.28964	1.87724E-08
	Worst	40.25456675	48.02336319	0.09737116	4583252.205	0.240192154
	Mean	8.886192319	4.13473568	0.011985054	1560315.509	0.02981008
	STD	12.0496735	11.2745274	0.025838424	994963.699	0.062183659
	MFE	30000	30000	30000	30238.76667	30035
14	Best	0.998003838	0.998003838	0.998003839	0.998055928	0.998003838
	Worst	0.998003838	0.998004194	0.999925881	3.968250346	0.998003843
	Mean	0.998003838	0.998003852	0.998257477	1.808262855	0.998003838
	STD	1.23698E-16	6.49804E-08	0.000527552	0.810513825	9.98569E-10
	MFE	30000	30000	30000	30199.9	30035.2
15	Best	0.000307486	0.000307486	0.000324243	0.001364568	0.000424113
	Worst	0.020434946	0.008333703	0.001272374	0.009562903	0.001380486
	Mean	0.00454563	0.001289295	0.000596688	0.004038841	0.000701339
	STD	0.008058756	0.001977607	0.000244477	0.002218998	0.000213724
	MFE	30000	30000	30000	30297.76667	30030
16	Best	-1.031628054	-1.031628233	-1.03162617	-1.031552471	-1.031627676
	Worst	-1.031584914	-1.03155237	-1.031600346	-1.011904581	-1.031602254

Table 3 Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for 23 classic benchmark functions

ID	Metric	Rao-1	Rao-2	Rao-3	FA-CL	ERA
	Mean	-1.031611222	-1.031611907	-1.031611279	-1.028544595	-1.031615743
	STD	1.08599E-05	1.47749E-05	7.75488E-06	0.003767043	8.22898E-06
	MFE	7041.333333	7202	8500	30176.5	2338
17	Best	0.397894345	0.397887438	0.397897956	0.397910357	0.397888025
	Worst	0.397999462	0.397996151	0.397988112	0.457108975	0.397998161
	Mean	0.397945174	0.397940169	0.39794788	0.407336514	0.397952448
	STD	2.93566E-05	3.44684E-05	2.58238E-05	0.012636874	3.35746E-05
	MFE	595.3333333	465.3333333	896	29274.23333	1524
18	Best	3	3	3.00001234	3.00144078	3
	Worst	3	3	3.002994418	3.461306359	3
	Mean	3	3	3.00043706	3.132162867	3
	STD	1.90941E-14	1.4162E-14	0.000581165	0.111886989	2.35699E-13
	MFE	2750.666667	6644	30000	30283.56667	13147.6
19	Best	-3.862647264	-3.862646836	-3.862630337	-3.86273971	-3.862476872
	Worst	-3.860015745	-3.860014013	-3.860166138	-3.814862203	-3.860018537
	Mean	-3.861284579	-3.860875224	-3.861230723	-3.847444698	-3.861305717
	STD	0.000759734	0.000609944	0.000769165	0.01227268	0.000744026
	MFE	526	326.6666667	721.3333333	29257.36667	1496
20	Best	-3.321514906	-3.321517556	-3.321340804	-3.232776201	-3.3216568
	Worst	-3.190272286	-3.20310205	-3.20310205	-2.774548607	-3.18590451
	Mean	-3.271481418	-3.27357853	-3.257887186	-2.964897304	-3.283422687
	STD	0.058118203	0.058528253	0.059569551	0.119154691	0.056560408
	MFE	15438	12787.33333	16832	30262.93333	14456.4
21	Best	-10.15319968	-10.15319968	-10.15319968	-9.237961427	-10.15319968
	Worst	-4.051730311	-2.630471668	-2.630471668	-2.348276139	-3.873011974
	Mean	-7.571532266	-7.286516369	-7.988655139	-5.13102673	-8.758677601
	STD	2.183528248	2.791706593	2.300114085	2.319287801	2.257015113
	MFE	30000	30000	30000	30201.03333	30029.6
22	Best	-10.40293072	-10.40293811	-10.40293612	-10.26936583	-10.40292495
	Worst	-3.724300347	-1.837592971	-7.655316059	-2.356385661	-4.785539658
	Mean	-8.513729479	-9.193001948	-10.14971362	-5.773917362	-9.962990885
	STD	2.516505404	2.672409474	0.667381659	2.772503044	1.251621589
	MFE	19810	7462.666667	10530.66667	30195.36667	28090
23	Best	-10.53640962	-10.53640895	-10.53640895	-9.998537379	-10.53640573
	Worst	-5.032711076	-2.421734027	-2.4273352	-2.420451607	-3.835426802
	Mean	-9.76293601	-8.189952935	-9.457598582	-5.207676489	-10.10837656

Table 3 Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for 23 classic benchmark functions

ID	Metric	Rao-1	Rao-2	Rao-3	FA-CL	ERA
	STD	1.635492849	3.653678401	2.385255584	2.37062617	1.331195279
	MFE	18558.66667	10478	11730.66667	30157.76667	27428.6
	FMR	2.43	3.17	2.13	4.83	1.52
	Rank	3	4	2	5	1

As a summary, based on Table 3, ERA reaches better mean solutions than all the competitors for 10 benchmark functions. It gives the same and worse mean solutions for 4 and 9 benchmark functions, respectively. Statistically, based on the Friedman mean rank (FMR), ERA gives the highest performance with the lowest FMR of 1.52. The Wilcoxon rank-sum test (WRST) illustrated in Table 4 confirms that ERA is significantly better than all the competitors for the six benchmark functions (ID = 4, 7, 9, 10, 12, and 23), where all the p-values are less than 0.05. Meanwhile, for the four benchmark functions (ID = 5, 6, 20, and 21), ERA is only significantly better than some competitors but not for the others.

Moreover, the detailed investigations are then provided by the convergence curve analysis. The three subsections below discuss the convergence curves in detail for three benchmark groups: high-dimensional unimodal, high-dimensional multimodal, and low-dimensional multimodal.

3.3.1. Investigation on 30-dimensions unimodal functions

A detailed investigation of the seven 30-dimensions unimodal benchmark functions, ID = 1 to 7, is discussed by illustrating two convergence analyses of the proposed ERA and all the competitors. For each benchmark function, the maximum number of function evaluations is set to 30,000 with 30 runs to reduce the coincidence.

Figure 6 shows the evolution of all the algorithms until convergence to the optimum solution for the benchmark function of Sphere (ID = 1). The horizontal axis is the generation, calculated as 30,000 function evaluations divided by the

Table 4: The p-values of Wilcoxon rank sum test (WRST) for 23 classic benchmark functions

ID	ERA vs Rao-1	ERA vs Rao-2	ERA vs Rao-3	ERA vs FA-CL
1	3.68973E-11	3.01986E-11	3.01986E-11	3.01986E-11
2	8.89099E-10	3.01986E-11	3.01986E-11	3.01986E-11
3	3.01986E-11	3.01986E-11	3.01986E-11	3.01986E-11
4	3.01986E-11	3.01986E-11	4.57257E-09	3.01986E-11
5	0.027086318	0.115362360	0.000526404	3.01986E-11
6	5.29270E-12	0.002309997	0.537496020	5.18120E-12
7	3.33839E-11	3.33839E-11	0.022360148	3.01986E-11
8	0.000421751	0.105469947	0.065671258	3.01986E-11
9	3.01986E-11	3.01986E-11	3.01986E-11	3.01986E-11
10	3.01986E-11	3.01986E-11	8.48477E-09	3.01986E-11
11	8.66343E-05	2.01522E-08	0.501143668	3.01986E-11
12	0.040595001	1.28704E-09	3.83067E-05	3.01986E-11
13	0.009883401	0.013271805	0.001766564	3.01986E-11
14	0.405861585	9.89193E-09	6.4749E-120	5.21903E-12
15	0.318136088	0.529748183	0.074827008	3.33839E-11
16	0.093340797	0.420386330	0.055545693	3.01986E-11
17	0.437641335	0.157975689	0.510597937	4.19968E-10
18	0.036392066	0.812931300	3.01041E-11	3.01041E-11
19	0.935191970	0.022360148	0.641423523	1.42942E-08
20	0.369977675	0.110560585	0.659705270	7.38908E-11
21	0.283376373	0.425345373	0.578792661	4.44405E-07
22	0.923442132	0.000556012	5.97056E-05	1.28704E-09
23	0.003670893	0.019094054	0.001235991	4.19968E-10

population size p. The random seeds of the 30 initial populations are the same for the algorithms that use the same optimum population size p. Hence, in this case, Rao-1, Rao-2, and FA-CL use the same initial population since they have the same optimum $p = 20$. In contrast, the Rao-3 and ERA use a different initial population because they have the optimum $p = 40$ and $p = 60$, respectively. Due to the different optimum p for each algorithm, the evolution is illustrated using the different step sizes of generation to get fairness. Here, the proposed ERA uses a step size of 2, Rao-3 uses 3, and the rests use 6 so that all the algorithms show the same generations of 1 to 250. It can be seen in Figure 6 that the ERA is worse than Rao-1 and Rao-3. This result also applies to two other similar unimodal functions ID = 2 and 3.

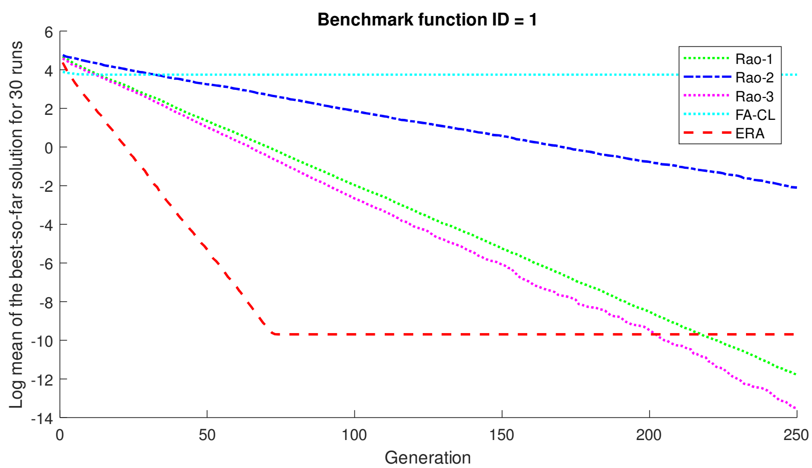


Figure 6: Convergence analysis for a unimodal benchmark function of Sphere (ID = 1)

Figure 7 shows the evolution of all the algorithms for the benchmark function of Schwefel 2.21 (ID = 4). ERA converges much faster than the others. It converges in the one-fourth of the evolution, and, at the end of evolution, it gives the lowest mean solution compared to Rao-1, Rao-2, Rao-3, and FA-CL that produce much worse solutions. Similar results also happen to three other unimodal functions ID = 5, 6, and 7.

3.3.2. Investigation on 30-dimensions multimodal functions

Next, a detailed investigation of the six 30-dimensional multimodal benchmark functions, ID = 8 to 13, is illustrated by two convergence analyses of the proposed ERA and all other algorithms. Figure 8 shows the convergence

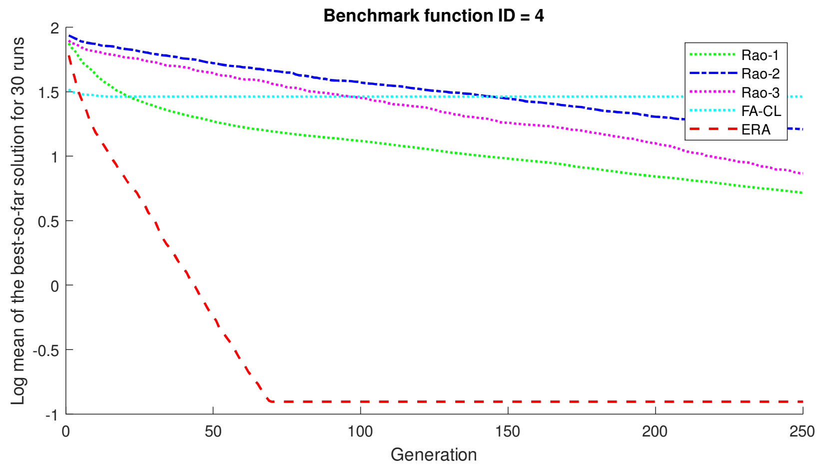


Figure 7: Convergence analysis for a unimodal benchmark function of Schwefel 2.21 (ID = 4)

curves for the multimodal function of Schwefel (ID = 8) that has many local minima. ERA performs a little worse than Rao-1 and Rao-3, where it converges to a slightly bigger solution. This result also applies to two other multimodal functions (ID = 11 and 13).

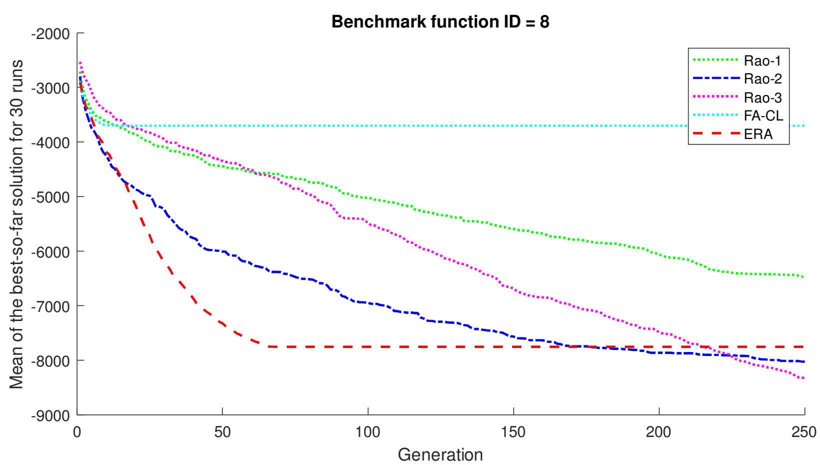


Figure 8: Convergence analysis for a multimodal benchmark function of Schwefel (ID = 8)

Next, the convergence analysis is provided for the multimodal function of Rastrigin with ID = 9 that also has many local minima. Figure 9 illustrates that the ERA converges much faster than the others. It evolves quickly in the beginning generations and gives the lowest mean solution among the competitors at the end of evolution. ERA also converges similarly for two other unimodal functions with ID = 10 and 12.

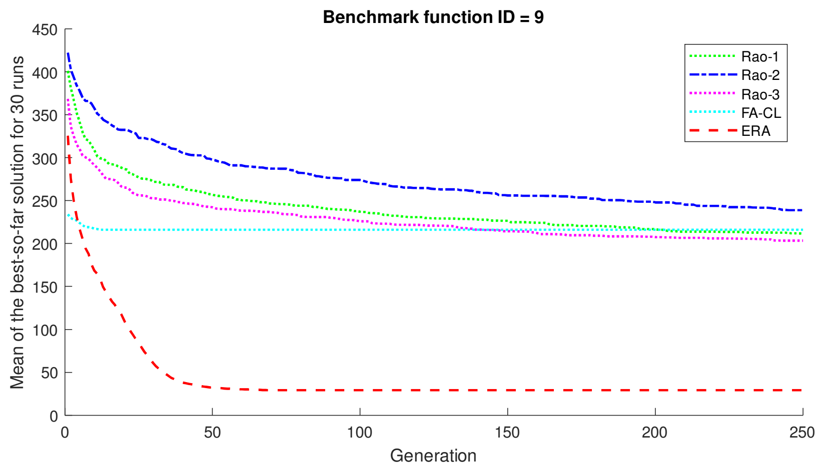


Figure 9: Convergence analysis for a multimodal benchmark function of Rastrigin (ID = 9)

3.3.3. Investigation on low-dimension multimodal functions

Finally, the detailed investigations of ten benchmark low-dimension multimodal functions, ID = 14 to 23, are also illustrated by some convergence analyses of ERA and the competitors. Figure 10 shows the evolution of all the algorithms for the 4-dimensions multimodal function of Shekel 7 (ID = 22) that has broad flat areas. In this case, ERA converges to a similar solution to Rao-3.

Furthermore, the convergence analysis is carried out for the 4-dimensions multimodal function of Shekel 10 (with ID = 23) with broad flat areas. Figure 11 shows that ERA performs the best evolution and converges to a better solution than the competitors. This result also applies to three other low-dimensions multimodal functions with ID = 14, 15, and 21. Meanwhile, ERA gives the same (or similar) convergence curves as the competitors for 16, 17, 18, 19, and 20.

Those results of FMR, WRST, and convergence curves indicate that ERA generally outperforms all the competitors. It proves that the proposed schemes: two sub-populations and evolutionary operators equipped with the adaptation procedure can effectively control the exploration and exploitation balance. The detailed investigations on the fitness-based adaptation scheme will be given in Subsection 3.6.

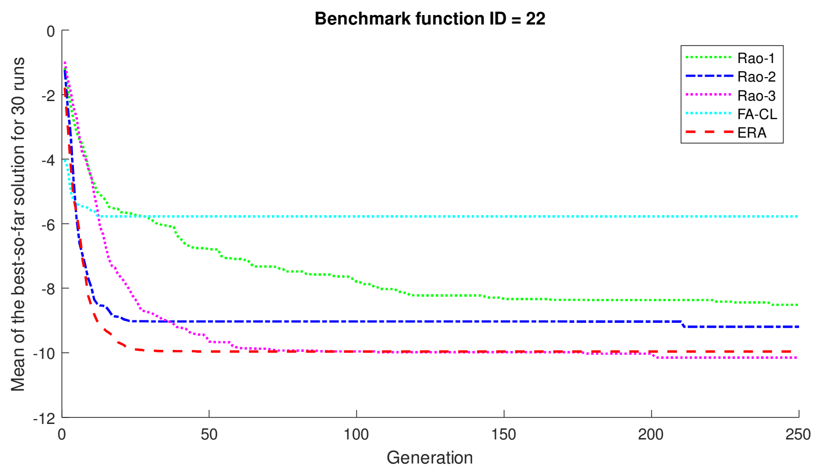


Figure 10: Convergence analysis for a multimodal benchmark function of Shekel 7 (ID = 22)

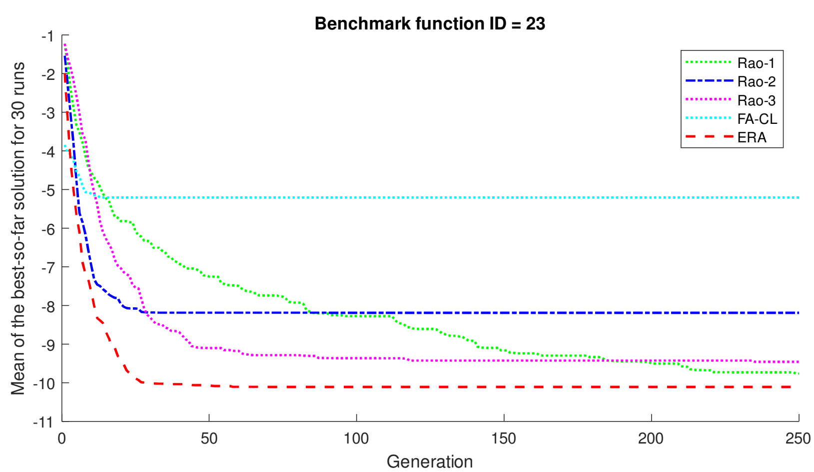


Figure 11: Convergence analysis for a multimodal benchmark function of Shekel 10 (ID = 23)

3.4. Evaluation on CEC-C06 2019

The CEC-C06 2019 is a set of ten modern benchmark functions, namely CEC01, CEC02, ..., CEC10. As described in [30], all the functions are scalable. The seven functions (CEC04 to CEC10) are shifted and rotated, but the others (CEC01 to CEC03) are not. Those seven functions are set as 10-dimensional minimization problems in the interval [-100, 100] while the rests have different dimensions of 9, 16, and 18 in the interval [-8192, 8192], [-16384, 16384], and [-4, 4], respectively. Besides, all ten benchmarks have the same global optimum of 1.

The proposed ERA is evaluated using those ten CEC-C06 2019 benchmarks, where their Matlab codes refer to [30], to see its ability to handle the effects of shift and rotation of the test functions. It is also compared with the four other algorithms in their best performances using the parameter settings described in Subsection 3.2. All algorithms are run 30 times with 30,000 function evaluations each to get meaningful statistical results. Moreover, both FMR and WRST with the p-values are also provided.

The experimental results illustrated in Table 5 show that ERA outperforms all the competitors for most benchmark functions. It reaches significantly better mean solutions for 7 out of 10 benchmarks: CEC01, CEC03, CEC05, CEC06, CEC07, CEC08, and CEC10. It gives little worse solutions than Rao-1 and Rao-2 for only three benchmarks: CEC02, CEC04, and CEC09. The Friedman mean rank shows that ERA is the first rank, where it achieves the lowest FMR of 1.50. Meanwhile Rao-1, Rao-2, Rao-3, and FA-CL give much worse FMR of 2.40, 2.50, 3.50, and 5.00, respectively.

Moreover, the Wilcoxon rank-sum test illustrated in Table 6 confirms that ERA is significantly better than all the competitors for the seven benchmark functions, where all the p-values are lower than the significance level of 0.05, except for the CEC03 where ERA is not significantly better than Rao-1. In contrast, for CEC02, ERA is slightly worse than Rao-1 with a p-value of bigger than 0.05. Meanwhile, for CEC04 and CEC09, ERA is much worse than Rao-1 and Rao-2 with p-values of less than 0.05.

Table 5 Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for ten benchmarks of CEC-C06 2019

ID	Metric	Rao-1	Rao-2	Rao-3	FA-CL	ERA
CEC01	Best	390354788.1	2916410938	1456657938	20718576008	25205057.57
	Worst	18579735758	26602967009	22606057356	3.52496E+11	4087331440
	Mean	2969561815	12158813001	8150847618	1.24685E+11	873534390.1
	STD	3458651870	6343181227	6050933807	88017205415	854049190.6
	MFE	30000	30000	30000	30258.9	30034.8
CEC02	Best	17.34285714	17.34285714	17.39624973	985.6431507	17.34385166
	Worst	17.34285714	17.34285714	17.46552571	3599.0291	17.38067398
	Mean	17.34285714	17.34285714	17.43100117	2309.760136	17.35587136
	STD	7.04391E-15	6.66287E-15	0.019863442	686.3489262	0.008533771
	MFE	30000	30000	30000	30167.6	30050
CEC03	Best	12.70240422	12.70240422	12.70240422	12.70243367	12.70240422
	Worst	12.70251646	12.70252446	12.70253809	12.70313897	12.70240457
	Mean	12.70241519	12.70242315	12.70243599	12.70275408	12.70240423
	STD	2.61665E-05	3.04861E-05	3.51307E-05	0.000177661	6.36292E-08
	MFE	30000	30000	30000	30194.93333	30036.4
CEC04	Best	28.09976484	30.65215071	161.5987987	1416.884608	12.6848588
	Worst	55.30958498	67.28669199	293.2794152	7138.324036	137.9261529
	Mean	39.24039894	48.18340414	214.1502774	3937.361268	42.30643978
	STD	6.336978985	8.897804689	35.66651224	1184.043609	30.04084127
	MFE	30000	30000	30000	30215.03333	30032.4
CEC05	Best	1.280029538	1.40996188	1.5745926	1.943716683	1.025782758
	Worst	1.698018753	2.006177027	1.97844018	2.898249478	1.401388266
	Mean	1.521217618	1.676196554	1.810647002	2.607524	1.152216958
	STD	0.129249904	0.115365294	0.087314766	0.244163606	0.089725781
	MFE	30000	30000	30000	30205.93333	30042
CEC06	Best	9.733872974	9.387891192	8.762132719	10.44553117	9.112980123
	Worst	11.45221129	11.28957571	11.75705625	13.24136925	10.88436736
	Mean	10.56696371	10.46733143	10.51740995	12.09261107	10.03315348
	STD	0.440170202	0.509698432	0.686756457	0.804138499	0.496448922
	MFE	30000	30000	30000	30177.5	30040.4
CEC07	Best	272.7315304	155.7756802	286.8881473	349.6263842	155.6301339
	Worst	909.7970404	873.1138007	971.9938573	1424.750579	623.6837399
	Mean	621.4329426	523.9535639	619.1594318	982.5206281	397.7660801
	STD	182.4620119	177.2735789	175.8854884	241.0657041	115.7688378
	MFE	30000	30000	30000	30189.7	30055.6
CEC08	Best	5.3786702	5.158716685	4.494612984	5.437849384	2.610104398

Table 5 Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for ten benchmark of CEC-C06 2019

ID	**Metric**	**Rao-1**	**Rao-2**	**Rao-3**	**FA-CL**	**ERA**
	Worst	6.875548609	6.47014268	6.121359357	7.20917994	5.372692569
	Mean	6.000536425	5.812160004	5.518584178	6.554942645	4.300037284
	STD	0.355903577	0.33238346	0.394227163	0.415653324	0.742690621
	MFE	30000	30000	30000	30155.36667	30048.2
CEC09	Best	2.344511638	2.410388957	8.415440069	107.8777016	2.471916423
	Worst	2.364279095	2.678458436	106.5608922	876.6191415	18.25738167
	Mean	2.352775129	2.519079794	47.78092232	560.843819	4.325248722
	STD	0.004691204	0.063901687	21.70671343	192.3434718	2.804045367
	MFE	30000	30000	30000	30187.13333	30044.2
CEC10	Best	20.14320415	20.24340503	20.12140073	20.09700221	20.14872943
	Worst	20.57898412	20.52293164	20.59178469	20.77386591	20.49628839
	Mean	20.42696562	20.40786804	20.42908964	20.60276541	20.34605205
	STD	0.082810151	0.073689655	0.085669202	0.131641593	0.072081928
	MFE	30000	30000	30000	30155.63333	30027.6
	FMR	2.40	2.50	3.50	5.00	1.50
	Rank	2	3	4	5	1

Furthermore, the detailed investigations are then discussed by illustrating the convergence analysis of ERA and all the competitors. For each benchmark function, the maximum number of function evaluations is set to 30,000 with 30 runs. Figure 12 illustrates the evolutionary processes of all algorithms until converge the optimum solution for CEC01. In this case, ERA converges to a much better solution than the other algorithms. ERA also gives similar curves for six other benchmarks: CEC03, CEC05, CEC06, CEC07, CEC08, and CEC10. Figure 13 and 14 illustrate the converge curves for CEC05 and CEC08. Impressively, for CEC05, ERA evolves most quickly in the beginning generations and finally gives the best mean solution of 1.152216958, which is quite close to the known global optimum of 1.

Next, the detailed investigation is then carried out for CEC02. Figure 15 shows that ERA converges to a similar solution to Rao-1, Rao-2, and Rao-3. In addition, ERA also gives similar curves for two other benchmarks: CEC04 and

Table 6: The p-values of Wilcoxon rank sum test (WRST) for ten benchmark functions of CEC-C06 2019

ID	ERA vs Rao-1	ERA vs Rao-2	ERA vs Rao-3	ERA vs FA-CL
CEC01	1.33668E-05	3.68973E-11	6.72195E-10	3.01986E-11
CEC02	2.36384E-12	6.31878E-12	3.01986E-11	3.01986E-11
CEC03	0.065461305	1.01761E-05	1.77301E-09	3.01986E-11
CEC04	0.053685253	0.004856016	3.01986E-11	3.01986E-11
CEC05	8.99341E-11	3.01986E-11	3.01986E-11	3.01986E-11
CEC06	0.000212646	0.002156638	0.001370333	9.91863E-11
CEC07	1.09069E-05	0.005322078	4.7445E-06	2.37147E-10
CEC08	3.01986E-11	7.38908E-11	4.99795E-09	3.01986E-11
CEC09	3.01986E-11	4.19968E-10	3.68973E-11	3.01986E-11
CEC10	5.60728E-05	0.00185748	3.59234E-05	7.38029E-10

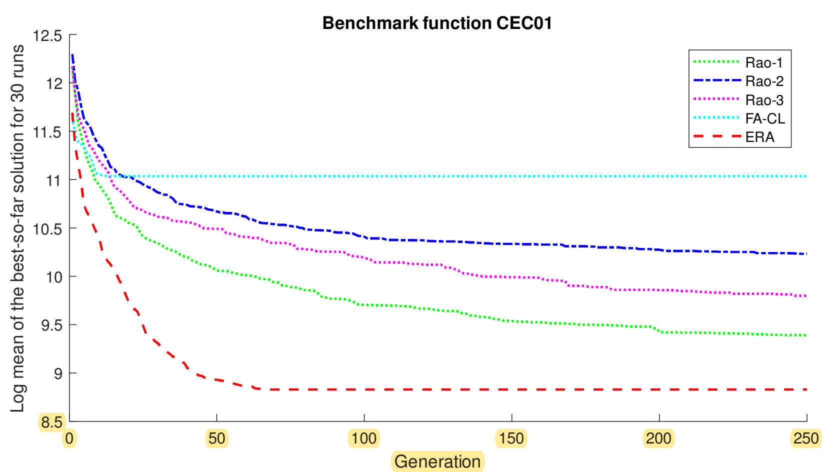


Figure 12: Convergence analysis for CEC01

CEC09, as illustrated in Figure 16 and Figure 17.

The results of FMR, WRST, and convergence curves above indicate that ERA is generally better than all the competitors in handling the effects of shift and rotation. It can be achieved since ERA is designed with two schemes:

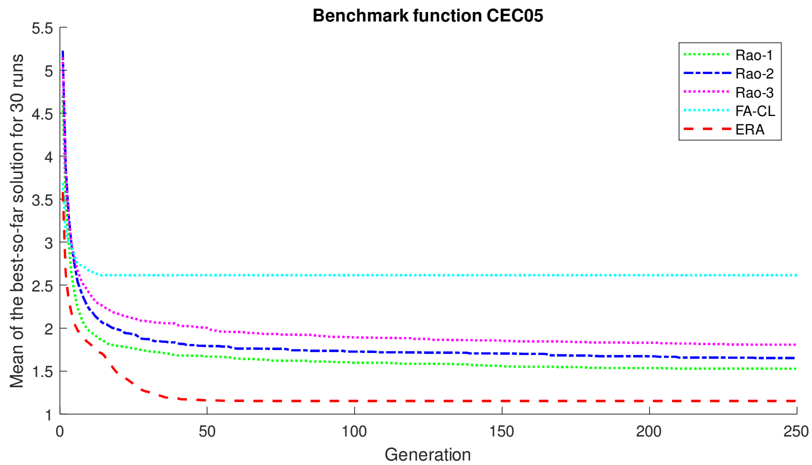


Figure 13: Convergence analysis for CEC05

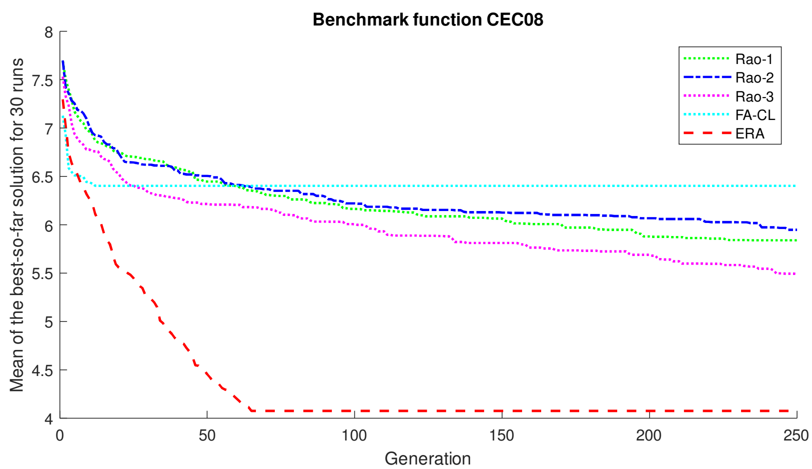


Figure 14: Convergence analysis for CEC08

two sub-populations and evolutionary operators. Besides, it is also equipped with a fitness-based adaptation scheme to dynamically tune the three sensitive parameters: s, a, and b throughout the evolution process, which effectively controls the exploration and exploitation balance in searching for the global optimum solution. A detailed investigation will be provided in Subsection 3.6.

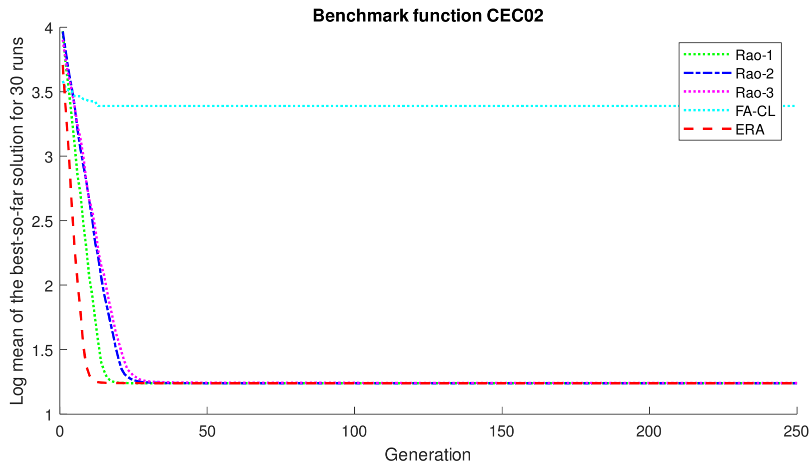


Figure 15: Convergence analysis for CEC02

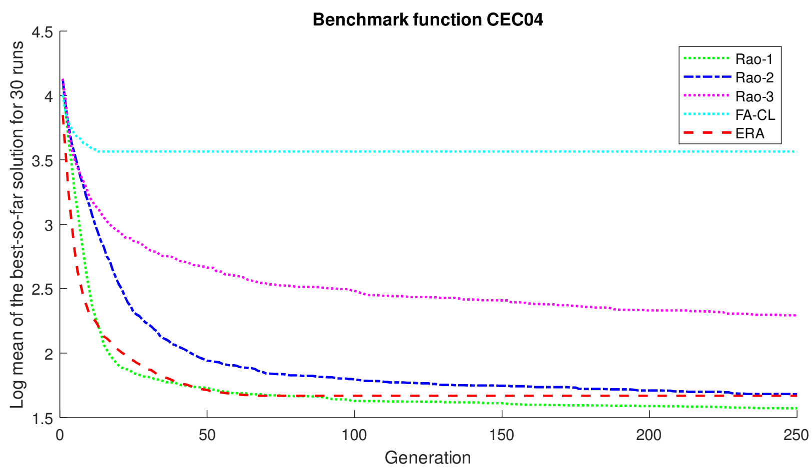


Figure 16: Convergence analysis for CEC02

3.5. Evaluation on real world problems

Finally, ERA is evaluated using the global trajectory optimization problems (GTOP), which European Space Agency provides. Here, five real-world problems: Cassini1, GTOC1, Messenger, Sagas, and Cassini2 are used as the benchmarks to examine its capability to tackle the constrained problems. Here, the four cases: Cassini1, Messenger, Sagas, and Cassini2 are the minimization problems while GTOC1 is a maximization. Here, the summary of those five

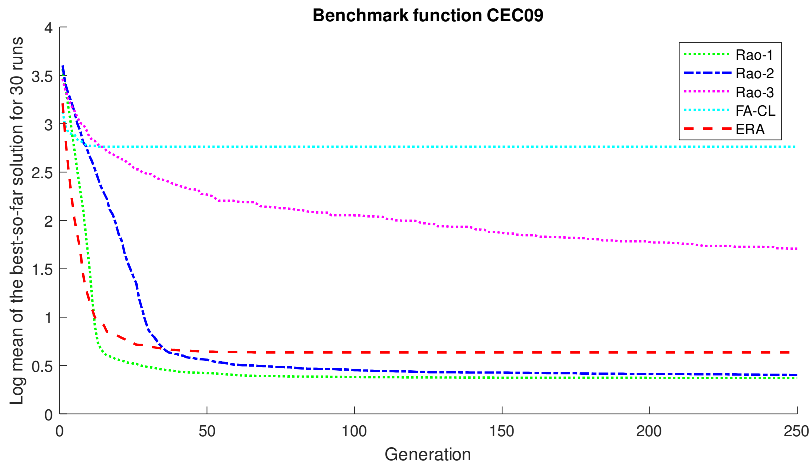


Figure 17: Convergence analysis for CEC02

problems is given briefly; the more detailed descriptions and their Matlab codes can be seen in [31, 33, 34].

As described in [31], Cassini1 is a mission of multiple gravity assist (MGA) without the possibility of using deep space manouvres, which is related to the Cassini spacecraft trajectory design problem. The objective is to minimize the total delta velocity $\triangle V$ accumulated during the mission with some given constraints, where the planetary fly-by sequence is Earth-Venus-Venus-Earth-Jupiter-Saturn. It has six dimensions with a known global minimum solution of 4.93 km/sec.

GTOC1 is also an MGA problem, where the objective is to maximize the change in the semi-major axis of the asteroid orbit following an anaelastic impact of the spacecraft with the asteroid under several given constraints. It has 8 dimensions and the known global maximum is $f_{max} = 1,580,599$ kg km^2/sec^2. In this paper, GTOC1 is converted into a minimization problem by modifying the objective function to be $f_{min} = 1,580,599 - f_{max} = 0$ to make it the same as all the other problems and simpler in the further evaluation.

Furthermore, Messenger, Sagas, and Cassini2 are the MGA problems with the possibility of using deep space manouvres (MGA-1DSM). The objective of

Messenger is to minimize the total delta velocity $\triangle V$. It has 18 dimensions, and the global minimum solution is 8.70257 km/sec. Meanwhile, the objective of Sagas is to minimize the overall mission length of fly-by Jupiter and reach 50AU. It is a 12-dimensional problem with a global minimum of 18.1923 years. Finally, Cassini2 is similar to Cassini1, but it is a bigger 22-dimensional problem with much higher complexity. The known best solution is 8.92401 km/sec.

In the five real-world problems, ERA is also compared with the four other algorithms in their best performances using the parameter settings described in Subsection 3.2. All algorithms are run 30 times with 200,000 function evaluations each as used in [31]. In addition, both FMR and WRST with the p-values are also provided to confirm the significance of their performance.

The experimental results illustrated in Table 7 show that ERA outperforms the competitors for three out of five problems. Hence, the Friedman mean rank places ERA in the first rank with the lowest FMR of 1.80. Unfortunately, the Wilcoxon rank-sum test illustrated in Table 8 indicates that ERA significantly outperforms (with p-values of less than 0.05) some of the competitors. It is slightly (not significantly) better than Rao-1, Rao-2, and Rao-3 for Cassini1, GTOC1, and Sagas, respectively.

Table 7 Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for global trajectory optimization problems

Problem	**Metric**	**Rao-1**	**Rao-2**	**Rao-3**	**FA-CL**	**ERA**
Cassini1	Best	5.303768392	4.936855136	5.61179549	7.074851303	5.612797557
	Worst	20.07805387	25.93297419	15.77360635	26.20831412	14.83107214
	Mean	11.87916323	12.12477947	12.56694735	16.6899857	10.61999692
	STD	3.554957586	3.64432772	2.23350672	4.840145141	2.876504936
	MFE	200000	200000	200000	200187.4333	200044.6
GTOC1	Best	631827.7829	458914.6802	376896.2971	1411461.987	710462.7745
	Worst	1548886.166	1551541.426	1536925.137	1569631.301	1311152.824
	Mean	1147746.121	1093465.404	1051211.872	1524073.714	956084.4087
	STD	233788.8767	331123.7832	266223.3551	43045.94967	136868.8532
	MFE	200000	200000	200000	200124.6333	200049.6
Messenger	Best	12.41560806	11.27514945	11.99302003	20.78291121	14.33306793
	Worst	25.50492428	20.425443	22.54997489	28.24295896	20.46142743
	Mean	18.35040915	14.97590782	16.61688953	25.51273265	16.37566883

Table 7 Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for global trajectory optimization problems

Problem	**Metric**	**Rao-1**	**Rao-2**	**Rao-3**	**FA-CL**	**ERA**
	STD	3.328431397	2.476903152	2.802312585	2.076873921	1.412144879
	MFE	200000	200000	200000	200136.7	200047.8
Sagas	Best	76.02938156	322.8420001	246.9977338	1267.982722	504.8182077
	Worst	1615.462402	3734.488398	2310.454904	2319.048595	973.1650916
	Mean	1024.564265	1202.784813	1017.331876	1722.402349	927.9858521
	STD	345.5888343	619.2934569	310.6547919	205.0753697	99.79849535
	MFE	200000	200000	200000	200206.9	200045.6
Cassini2	Best	10.62971719	11.03729327	15.93387459	32.97381024	21.03860384
	Worst	44.68073676	36.96644582	32.11412822	47.76452256	30.01962572
	Mean	25.0919048	23.93646566	25.58562866	40.28520967	26.03675587
	STD	8.783109087	6.801655034	3.476676083	4.082917564	2.049424528
	MFE	200000	200000	200000	200088.5667	200036.6
	FMR	3.00	2.40	2.80	5.00	1.80
	Rank	4	2	3	5	1

Table 8: The p-values of Wilcoxon rank sum test (WRST) for five global trajectory optimization problems

Problem	ERA vs Rao-1	ERA vs Rao-2	ERA vs Rao-3	ERA vs FA-CL
Cassini1	0.318304227	0.072445596	0.00033679	8.1975E-07
GTOC1	0.000356384	0.108689773	0.02812867	3.01986E-11
Messenger	0.005084222	0.000691252	0.437641335	3.01986E-11
Sagas	0.022360148	0.000168132	0.111986872	3.01986E-11
Cassini2	0.332854692	0.008314609	0.529782491	3.01986E-11

Furthermore, the detailed investigations are then discussed by illustrating the convergence curves of ERA and the competitors. Figure 18 illustrates the evolution of all algorithms until to the optimum solution for CEC01. In this case, ERA converges to a better solution than the other algorithms. Like all the competitors, ERA evolves quickly in the beginning generations and gets stagnation. In addition, ERA also gives similar curves for GTOC1 and Sagas, as illustrated in Figure 19 and 21. However, for GTOC1, ERA reaches a much better solution.

For both Messenger and Cassini2 problems, ERA gives worse convergence curves than the Rao algorithms. Figure 20 informs that, in the beginning, ERA gives the same converge curve as the Rao-2, but it gets stuck after half of the generations while Rao-2 keeps evolving and reaches a little better solution. Meanwhile, Figure 22 shows that ERA evolves quickly in the early generations but finally converges to a slightly worse solution than the three Rao algorithms.

However, those results of FMR, WRST, and convergence curves inform that ERA is better than the competitors in tackling the constrained real-world problems. It can be implied that the two proposed schemes: two sub-populations and evolutionary operators, as well as the introduced adaptation procedure, effectively balance the exploration and exploitation strategies. A detailed investigation regarding the adaptation scheme will be provided in Subsection 3.6.

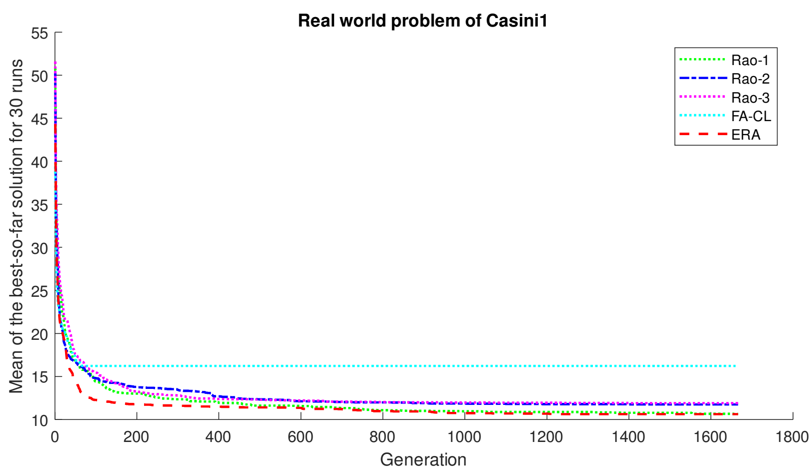


Figure 18: Convergence analysis for Cassini1

3.6. Investigation on fitness-based adaptation scheme

The proposed fitness-based adaptation scheme is evaluated here using some benchmarks to see its ability to control the exploration-exploitation balance.

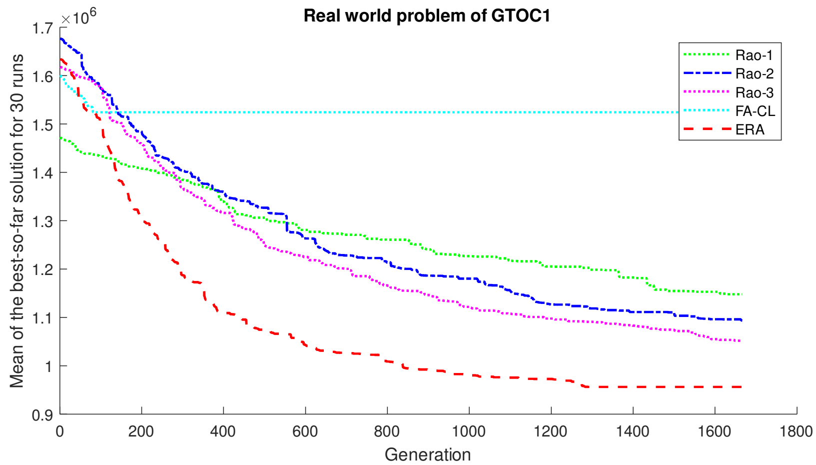


Figure 19: Convergence analysis for GTOC1

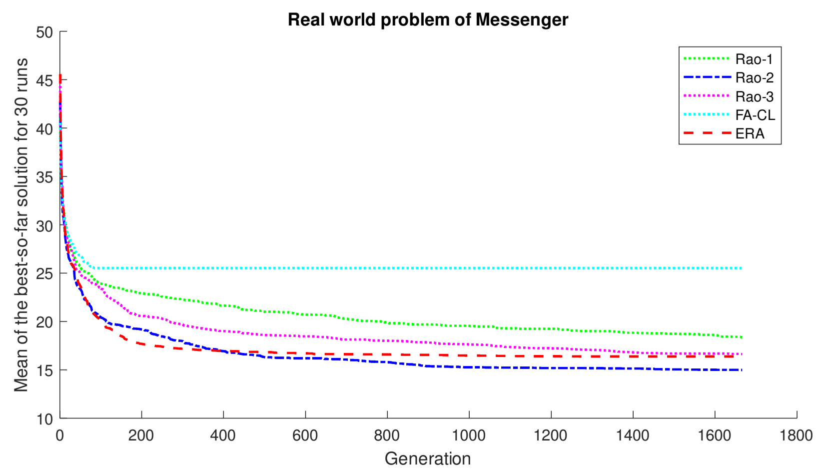


Figure 20: Convergence analysis for Messenger

First, two classic benchmarks (with ID = 1 and 7) are selected as the representative 30-dimensional unimodal functions. Figure 23a illustrates that, for the Sphere function (ID = 1) that has only one global optimum, ERA converges quite fast. It can be achieved since ERA works in an exploration manner in the beginning generations and then quickly changes into an exploitation fashion, where the three parameters s, a, and b reach around the maximum values of 0.9, 0.9, and 0.5, respectively, as shown in Figure 23b. Meanwhile, Figure 23c

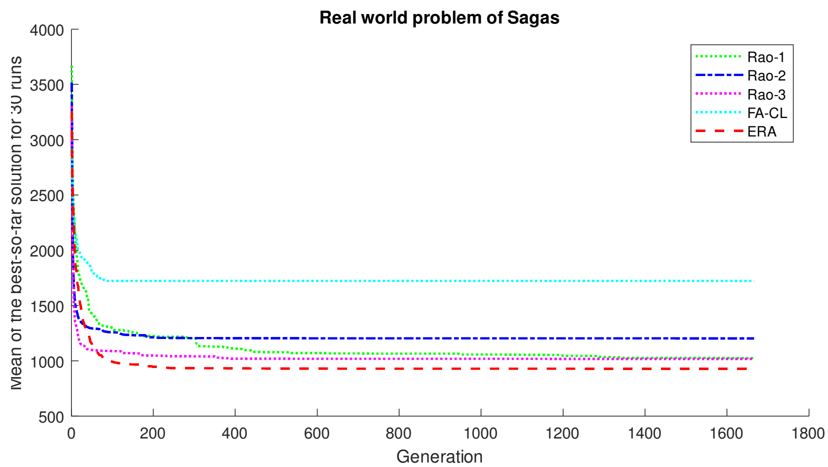


Figure 21: Convergence analysis for Sagas

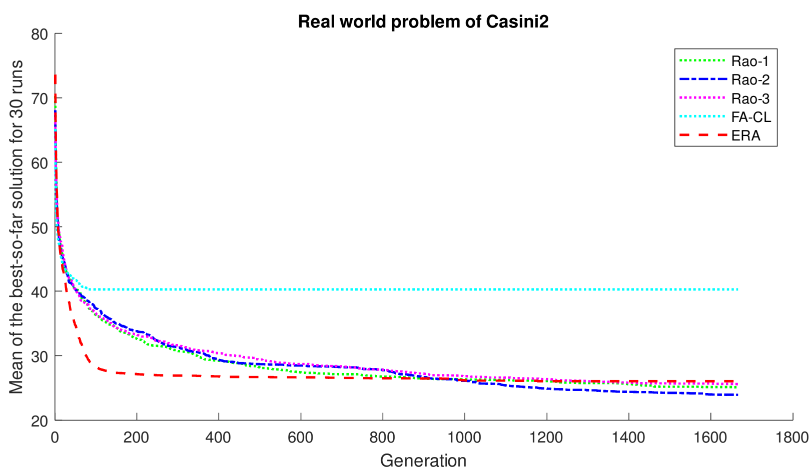
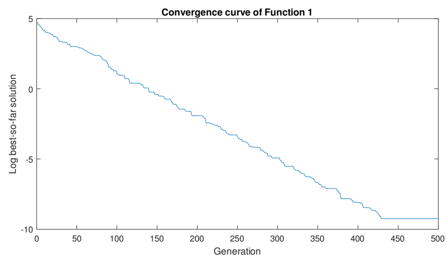


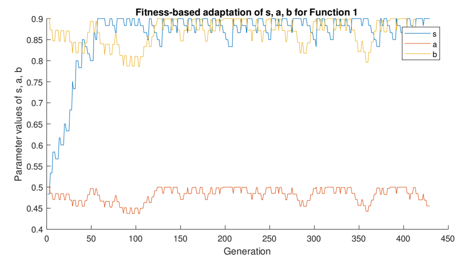
Figure 22: Convergence analysis for Cassini2

illustrates that, for the Quartic function (with ID = 7) having many noises, ERA converges more slowly and gets a stagnation. In this case, ERA works in an exploration strategy throughout the evolution, where s tends to go to the minimum value of 0.1 but a and b reach the maximum values of 0.9 and 0.5, respectively, as shown in Figure 23d.

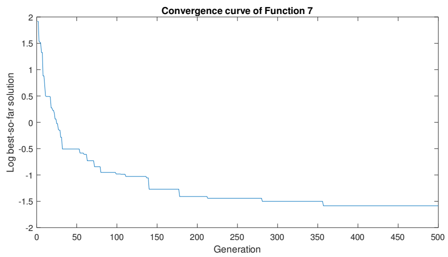
Next, two classic benchmarks with ID = 8 and 10 are selected as the representative 30-dimensional multimodal functions. Figure 24a illustrates that, for



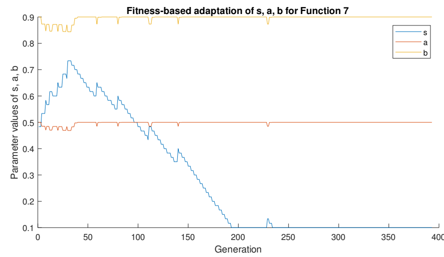
(a) Convergence curve of Function 1



(b) Adaptation curve of Function 1



(c) Convergence curve of Function 7

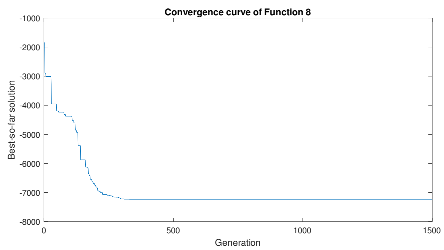


(d) Adaptation curve of Function 7

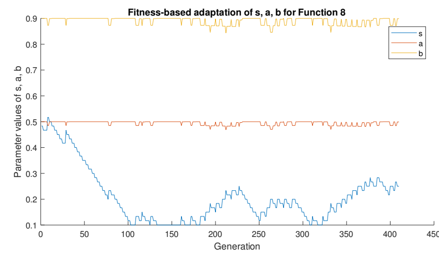
Figure 23: Convergence and adaptation curves for 30-dimensional unimodal benchmark functions with ID = 1 and 7

the Schwefel function (ID = 8) having many global optimum solutions, ERA evolves quickly in early generations but finally gets stuck on a local optimum. Here, ERA works in an exploration strategy throughout the evolution, where s tends to go around the minimum value but a and b on the maximum values, as shown in Figure 24b. Meanwhile, Figure 24c illustrates that, for the Ackley function (ID = 10) that also has many global optimum solutions, ERA converges quickly and also gets a stagnation. In this case, ERA tends to work in an exploitation strategy throughout the evolution, where s is on the maximum value, but a and b tends on the medium values, as shown in Figure 24d.

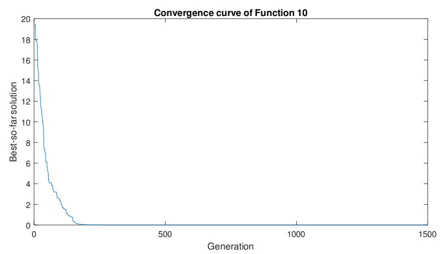
Two classic benchmarks with ID = 20 and 21 are then chosen as the representative low-dimensional multimodal functions. Figure 25a illustrates that, for the function of Hartman 6 (ID = 20) with many global optima, ERA converges quite fast to the global optimum. It can be seen that ERA works in an exploration-exploitation balance strategy throughout the evolution, where s



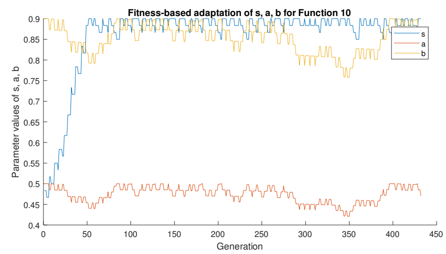
(a) Convergence curve of Function 8



(b) Adaptation curve of Function 8



(c) Convergence curve of Function 10

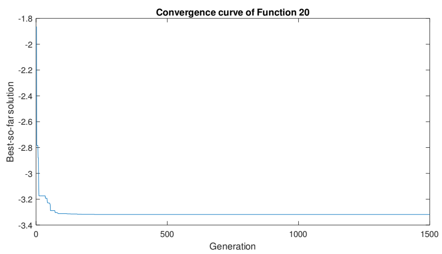


(d) Adaptation curve of Function 10

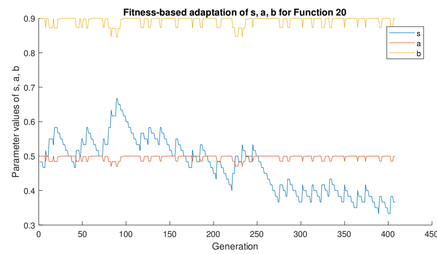
Figure 24: Convergence and adaptation curves for 30-dimensional multimodal benchmark functions: 8 and 10

is fluctuating between the minimum and the medium values while a and b are around the maximum values, as shown in Figure 25b. Meanwhile, Figure 25c illustrates that, for the function of Shekel 5 (ID = 21), ERA converges quickly to the global optimum. It tends to work in an exploration strategy throughout the evolution since Shekel 5 has a broad flat area, where s is around the minimum value, but a and b tends on the maximum values, as shown in Figure 25d.

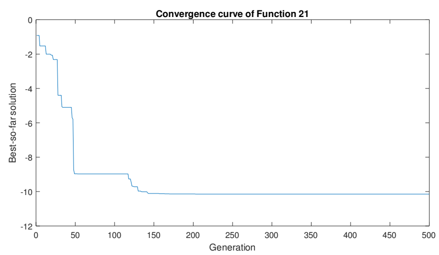
Two benchmarks of CEC01 and CEC05 are then chosen as the representative functions without and with both shifting and rotation, respectively. Figure 26a illustrates that, for the function without shifting and rotation, ERA evolves slowly. Unfortunately, it gets stuck for some generations and converges to a solution far from the global optimum. It can be seen that ERA works in an exploration-exploitation balance strategy throughout the evolution, where s is fluctuating between the minimum and the medium values while a and b are around the maximum values, as shown in Figure 26b. Meanwhile, Figure 26c



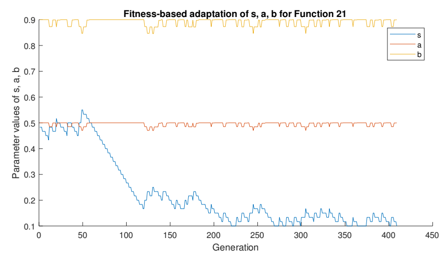
(a) Convergence curve of Function 20



(b) Adaptation curve of Function 20



(c) Convergence curve of Function 21

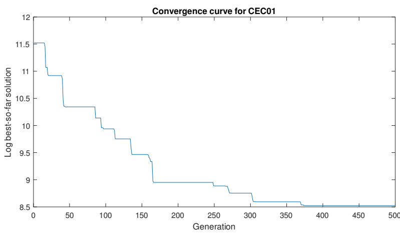


(d) Adaptation curve of Function 21

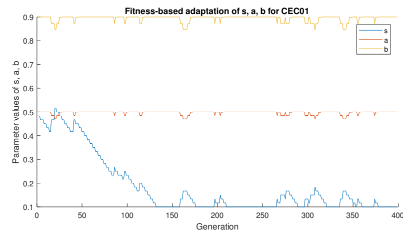
Figure 25: Convergence and adaptation curves for low-dimensional multimodal benchmark functions: 20 and 21

illustrates that, for the shifted and rotated function, ERA converges quickly to a solution near the global optimum of 1. It works dynamically in exploration and exploitation strategy during the evolution, where s is around the medium value, but a and b tends on the maximum values, as shown in Figure 26d.

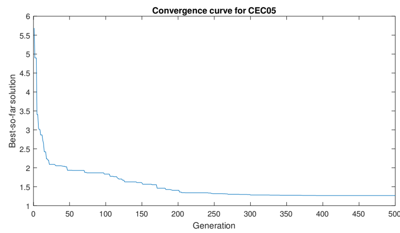
Finally, both Cassini1 and Messenger are selected as the representative of the real-world problems. Figure 27a illustrates that ERA evolves quickly in the beginning generations, but it gets stuck and converges to a solution far from the global optimum. It can be seen that ERA tends to work in an exploration manner throughout the evolution, where s is on the minimum value while a and b are around the maximum values, as shown in Figure 27b. Meanwhile, Figure 27c shows that ERA evolves slowly and converges to a solution near the global optimum. It works dynamically in exploration and exploitation strategy during the evolution, where s is fluctuating between the minimum and the medium values, but a and b tends on the maximum values, as shown in Figure 27d.



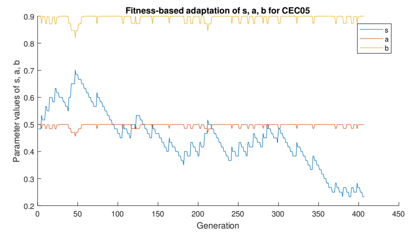
(a) Convergence curve of CEC01



(b) Adaptation curve of CEC01

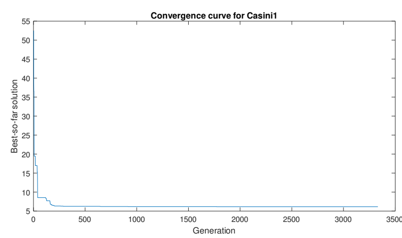


(c) Convergence curve of CEC05

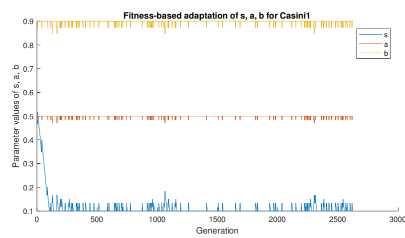


(d) Adaptation curve of CEC05

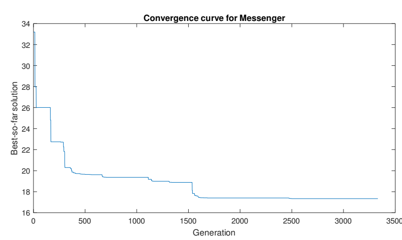
Figure 26: Convergence and adaptation curves for CEC01 and CEC05



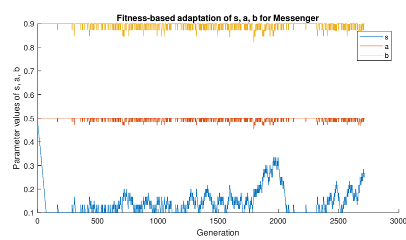
(a) Convergence curve of Cassini1



(b) Adaptation curve of Cassini1



(c) Convergence curve of Messenger



(d) Adaptation curve of Messenger

Figure 27: Convergence and adaptation curves for the global trajectory optimization problem of Cassini1 and Messenger

The convergence and adaptation curves above prove that the proposed adaptation scheme effectively controls the exploration and exploitation balance. This scheme makes ERA can handle many types of benchmark functions: unimodal, multimodal, shifted, rotated, and also real-world problems.

4. Conclusions

The proposed evolutionary Rao algorithm (ERA) works very well based on two additional schemes: splitting the population into two subpopulations based on their qualities: high and low, with a proper portion adaptively during the evolution, and exploiting two evolutionary operators: crossover and mutation. The evaluations of twenty-three classic benchmark functions and ten CEC-C06 2019 benchmarks show that it significantly outperforms all the competitors: Rao-1, Rao-2, Rao-3, and FA-CL, where it reaches the Friedman mean rank of 1.52 and 1.50, respectively, with the p-values of Wilcoxon rank-sum test of less than 0.05 for most of the benchmarks. Examining the five real-world global trajectory optimization problems inform that ERA gives significant performances only for some of the competitors. Detailed investigations prove that all the proposed schemes work well as they are designed and make ERA effectively control the exploration and exploitation balance. All the proposed schemes make ERA able to handle most of the benchmark functions with various types: unimodal, multimodal, shifted, rotated, and also real-world problems. However, in the future, a new advanced adaptation scheme to update the population size dynamically throughout the evolutionary process as well as a better mutation scheme will be created to improve the performance of ERA. Besides, it will be comprehensively examined using more challenging benchmarks.

References

[1] K. Hussain, M. N. Mohd Salleh, S. Cheng, Y. Shi, Metaheuristic research: a comprehensive survey, Artificial Intelligence Review 52 (4) (2019) 2191–2233. doi:10.1007/s10462-017-9605-z.

URL https://link.springer.com/article/10.1007/s10462-017-9605-z

[2] J. H. Holland, Genetic algorithms, Scientific American 267 (1) (1992) 66–72. doi:10.1038/scientificamerican0792-66.
URL https://www.scientificamerican.com/article/genetic-algorithms/

[3] A. Lambora, K. Gupta, K. Chopra, Genetic Algorithm- A Literature Review, in: 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), 2019, pp. 380–384. doi:10.1109/COMITCon.2019.8862255.

[4] F. Murillo, T. Neuenschwander, R. Dornberger, T. Hanne, Optimization of a Robotic Manipulation Path by an Evolution Strategy and Particle Swarm Optimization, in: ACM International Conference Proceeding Series, Association for Computing Machinery, 2020, pp. 36–41. doi:10.1145/3396474.3396488.

[5] Z. Li, Q. Zhang, Variable metric evolution strategies by mutation matrix adaptation, Information Sciences 541 (2020) 136–151. doi:10.1016/j.ins.2020.05.091.
URL https://doi.org/10.1016/j.ins.2020.05.091

[6] J. Cheng, Z. Pan, H. Liang, Z. Gao, J. Gao, Differential evolution algorithm with fitness and diversity ranking-based mutation operator, Swarm and Evolutionary Computation 61 (May 2020) (2021) 100816. doi:10.1016/j.swevo.2020.100816.
URL https://doi.org/10.1016/j.swevo.2020.100816

[7] R. Zhang, Z. Qiu, Optimizing hyper-parameters of neural networks with swarm intelligence: A novel framework for credit scoring, PLoS ONE 15 (6). doi:10.1371/journal.pone.0234254.

[8] V. Mp, B. Anand, Microprocessors and Microsystems Particle swarm optimization technique for multilevel inverters in solar harvesting micro grid system, Microprocessors and Microsystems 79 (August) (2020) 103288. doi:10.1016/j.micpro.2020.103288.
URL https://doi.org/10.1016/j.micpro.2020.103288

[9] D. Kumar, B. G. R. Gandhi, R. K. Bhattacharjya, Firefly Algorithm and Its Applications in Engineering Optimization, Modeling and Optimization in Science and Technologies 16 (2020) 93–103. doi:10.1007/978-3-030-26458-1_6.
URL https://www.springerprofessional.de/en/firefly-algorithm-and-its-applications-in-engineering-optimizati/17575198

[10] V. Kumar, D. Kumar, A Systematic Review on Firefly Algorithm: Past, Present, and Future, Archives of Computational Methods in Engineeringdoi:10.1007/s11831-020-09498-y.
URL https://www.x-mol.com/paper/1311351373626052608

[11] S. Gupta, K. Deep, Enhanced leadership-inspired grey wolf optimizer for global optimization problems, Engineering with Computers 36 (4) (2020) 1777–1800. doi:10.1007/s00366-019-00795-0.

[12] J. C. Bansal, S. Singh, A better exploration strategy in Grey Wolf Optimizer, Journal of Ambient Intelligence and Humanized Computingdoi:10.1007/s12652-020-02153-1.

[13] A. S. Assiri, A. G. Hussien, M. Amin, Ant lion optimization: Variants, hybrids, and applications, IEEE Access 8 (2020) 77746–77764. doi:10.1109/ACCESS.2020.2990338.
URL https://ieeexplore.ieee.org/abstract/document/9078091

[14] J. H. Holland, J. H. Holland, Others, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence, University of Michigan press, 1975.

[15] A. C. Rizal, S. Suyanto, Human-Like Constrained-Mating to Make Genetic Algorithm More Explorative, in: 2020 8th International Conference on Information and Communication Technology (ICoICT), IEEE, 2020, pp. 1–5. doi:https://doi.org/10.1109/ICoICT49345.2020.9166387.
URL https://ieeexplore.ieee.org/document/9166387/

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

[17] D. Li, W. Guo, A. Lerch, Y. Li, L. Wang, Q. Wu, An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization, Swarm and Evolutionary Computation 60 (December 2019) (2021) 100789. doi:10.1016/j.swevo.2020.100789.
URL https://doi.org/10.1016/j.swevo.2020.100789

[18] D. Sedighizadeh, E. Masehian, M. Sedighizadeh, H. Akbaripour, ScienceDirect GEPSO : A new generalized particle swarm optimization algorithm, Mathematics and Computers in Simulation 179 (2021) 194–212. doi:10.1016/j.matcom.2020.08.013.
URL https://doi.org/10.1016/j.matcom.2020.08.013

[19] X.-S. Yang, Firefly algorithms for multimodal optimization, in: Stochastic Algorithms: Foundations and Applications, Springer Berlin Heidelberg, 2009, pp. 169–178.

[20] I. Fister, I. Fister, X.-S. Yang, J. Brest, A comprehensive review of firefly algorithms, Swarm and Evolutionary Computation 13 (2013) 34–46.

[21] H. Peng, W. Zhu, C. Deng, Z. Wu, Enhancing firefly algorithm with courtship learning, Information Sciences 543 (2020) 18–42. doi:10.1016/j.ins.2020.05.111.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85088912963{&}doi=10.1016{%}2Fj.ins.2020.05.111{&}partnerID=40{&}md5=96db75459ce8599098979d9c83af6772

[22] S. Wu, Z. Wu, H. Peng, Enhancing Firefly Algorithm with Best Neighbor Guided Search Strategy, Wuhan University Journal of Natural Sciences 24 (6) (2019) 524–536. doi:10.1007/s11859-019-1432-4.

[23] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey Wolf Optimizer, Advances in Engineering Software 69 (2014) 46–61. doi:10.1016/j.advengsoft.2013.12.007.

[24] J. Luo, Z. Liu, Novel grey wolf optimization based on modified differential evolution for numerical function optimization, Applied Intelligence 50 (2) (2020) 468–486. doi:10.1007/s10489-019-01521-5.
URL https://link.springer.com/article/10.1007/s10489-019-01521-5

[25] S. Bahuguna, A. Pal, Annealed grey wolf optimization, Advances in Mathematics: Scientific Journal 9 (8) (2020) 5477–5489. doi:10.37418/amsj.9.8.18.

[26] W. Long, T. Wu, S. Cai, X. Liang, J. Jiao, M. Xu, A Novel Grey Wolf Optimizer Algorithm with Refraction Learning, IEEE Access 7 (2019) 57805–57819. doi:10.1109/ACCESS.2019.2910813.
URL https://ieeexplore.ieee.org/document/8689025

[27] M. H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, An improved grey wolf optimizer for solving engineering problems, Expert Systems with Applications 166. doi:10.1016/j.eswa.2020.113917.
URL https://www.sciencedirect.com/science/article/pii/S0957417420307107

[28] S. Mirjalili, Advances in Engineering Software The Ant Lion Optimizer, Advances in Engineering Software 83 (2015) 80–98. doi:10.1016/j.advengsoft.2015.01.010.
URL http://dx.doi.org/10.1016/j.advengsoft.2015.01.010

[29] R. V. Rao, Rao algorithms: Three metaphor-less simple algorithms for solving optimization problems, International Journal of Industrial Engineering Computations 11 (1) (2020) 107–130. doi:10.5267/j.ijiec.2019.6.002.

[30] J. M. Abdullah, Fitness Dependent Optimizer : Inspired by the Bee Swarming Reproductive Process, IEEE Access 7 (2019) 43473–43486. doi:10.1109/ACCESS.2019.2907012.

[31] T. Vinko, D. Izzo, Global Optimisation Heuristics and Test Problems for Preliminary Spacecraft Trajectory Design, Tech. Rep. GOHTPPSTD (2008).
URL http://www.esa.int/gsp/ACT/doc/INF/pub/ACT-TNT-INF-2008-GOHTPPSTD.pdf

[32] X. Xia, L. Gui, Y. Zhang, X. Xu, F. Yu, A fitness-based adaptive differential evolution algorithm, Information Sciences 549 (2021) 116–141. doi:10.1016/j.ins.2020.11.015.
URL https://doi.org/10.1016/j.ins.2020.11.015

[33] E. S. A. Advanced Concepts Team, Global Trajectory Optimisation Problems Database (2021).
URL https://www.esa.int/gsp/ACT/projects/gtop/

[34] T. Vinko, D. Izzo, C. Bombardelli, Benchmarking different global optimisation techniques for preliminary spacetrajectory design, in: 58th International Astronautical Congress, 2007.
URL http://www.esa.int/gsp/ACT/doc/MAD/pub/ACT-RPR-MAD-2007-BenchmarkingDifferentGlobalOptimisationTechniques.pdf

1. Suyanto received the B.Sc. on Informatics Engineering from STT Telkom (now Telkom University), Bandung, Indonesia in 1998, the M.Sc. on Complex Adaptive Systems from Chalmers University of Technology, Goteborg, Sweden, in 2006, and the Doctor on Computer Science from Universitas Gadjah Mada in 2016. Since 2000, he joined STT Telkom as a lecturer in School of Computing. His research interests include artificial intelligence, machine learning, deep learning, swarm intelligence, speech processing, and computational linguistics. Scopus ID: 56843751100, Researcher ID: AAB-5223-2021, Publon ID: 4171399, Orcid: https://orcid.org/0000-0002-8897-8091.

2. Agung Toto Wibowo received the B.Sc. on Informatics Engineering from STT Telkom (now Telkom University), Bandung, Indonesia in 2005, the M.T. on Electrical Engineering from Bandung Institute of Technology, Indonesia, in 2009, and the Doctor of Philosophy on Computing Science from University of Aberdeen in 2019. Since 2006, he joined IT Telkom as a lecturer in School of Computing. His research interests include artificial intelligence, machine learning, deep learning, swarm intelligence, speech processing, and recommender systems. Scopus ID: 57195616672

3. Said Al Faraby received M.Sc. in Artificial Intelligence from University of Amsterdam in 2015. Before that, he completed B.Sc. on Informatics Engineering from IT Telkom (now Telkom University), Bandung, Indonesia in 2010. Since 2015 he joined Telkom University as a lecturer at School of Computing. His research interests mainly in the area of machine learning, natural language processing, and intelligence systems. Scopus ID: 55845346500, Publon ID: 4214736.

4. Siti Saadah received the Bachelor and Master degree in Informatics Engineering from Telkom Institute of Technology (now Telkom University), Bandung, Indonesia in 2009 and 2012. Since 2009, she joined Telkom University as a lecturer in School of Computing. She is Teaching Design and Analysis Algorithm, Artificial Intelligence, Theory Authomata at Telkom University. Her research interests include machine learning, financial computing, AI healthcare, prediction and simulation. Scopus ID: 55523371300, Researcher ID: AAD-6187-2021, Publon ID: 4215578.

5. Rita Rismala received the Bachelor and Master degree in Informatics Engineering from Telkom Institute of Technology (now Telkom University), Bandung, Indonesia in 2010 and 2013. She is now a Ph.D student in School of Electrical Engineering and Informatics, Bandung Institute of Technology, Bandung, Indonesia. Since 2011, she joined Telkom University as a lecturer in School of Computing. Her research interests include machine learning, deep learning, recommender system, and natural language processing. Scopus ID: 55844928300, Researcher ID: AAE-4960-2021, Publon ID: 4187049.

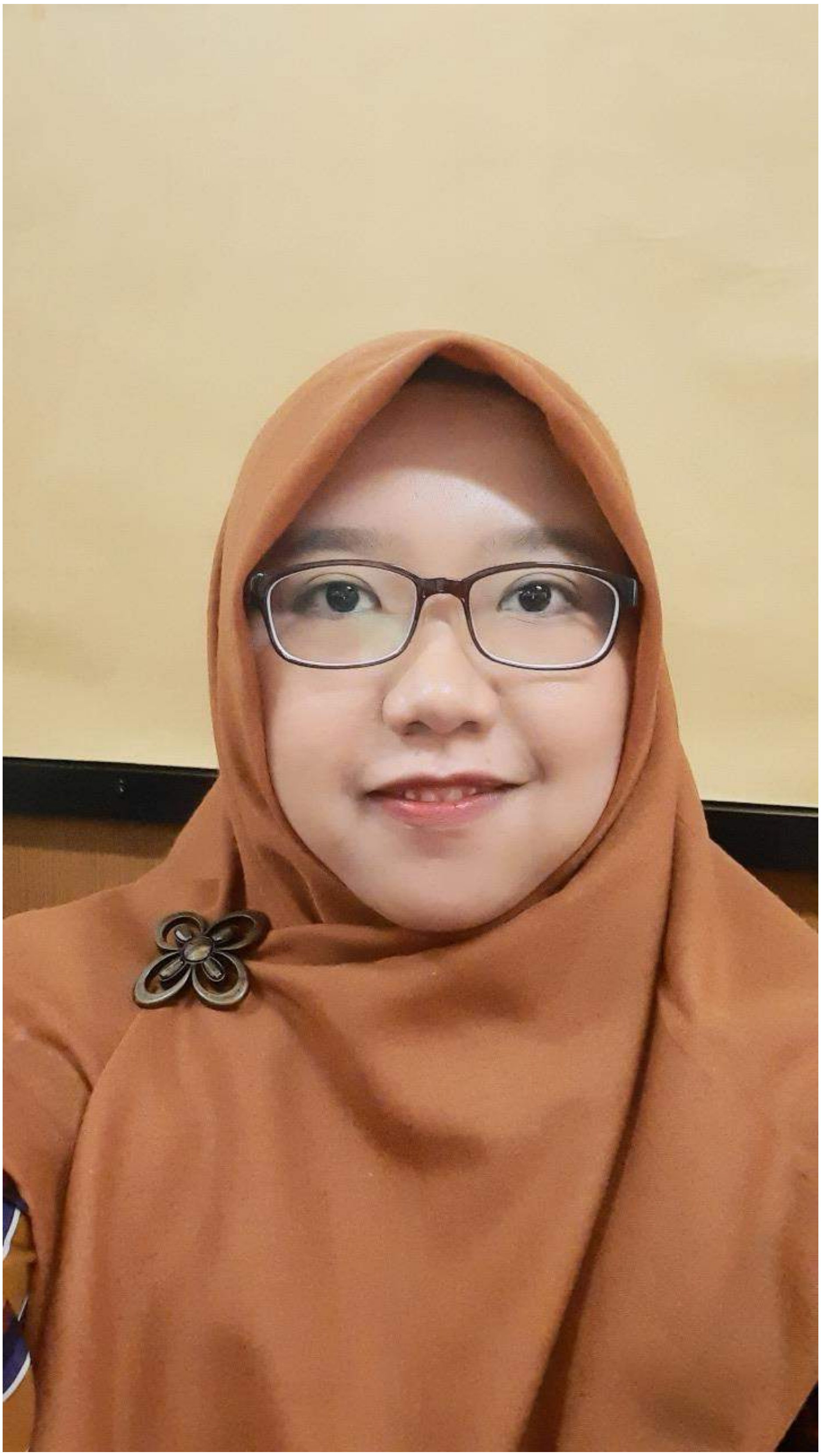
Author Photo

Suyanto Suyanto

Agung Toto Wibowo

Said Al Faraby

Siti Saadah



Rita Rismala

Declaration of interests

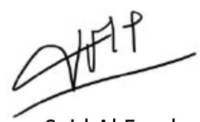
☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☒The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Authors,

Suyanto Suyanto

Agung Toto Wibowo



Said Al Faraby

Siti Saadah

Rita Rismala

Author Statement

We declare that we have no known competing financial interest or personal relationships that could have appeared to influence the work reported in the manuscript entitled “Evolutionary Rao Algorithm”.

The Credit author statement is as follow. **Suyanto Suyanto**: principal investigator, conceptualization, methodology, data curation, writing original draft preparation. **Agung Toto Wibowo**: literature review evolutionary algorithm, analysis, and editing, **Said Al Faraby:** literature review evolutionary algorithm approach, analysis Rao and editing, **Siti Saadah:** literature review genetic algorithm, analysis Evolutionary Rao and reviewing, **Rita Rismala:** analysis Rao result and reviewing.

March 11, 2021

Authors,

Suyanto Suyanto   Agung Toto Wibowo   Said Al Faraby   Siti Saadah   Rita Rismala

Evidence of correspondence

Evolutionary Rao Algorithm

1. First submission (04 January 2021)
2. LoA with Major Revision (23 February 2021)
3. Responses to Reviewers, Final submission (12 March 2021)
4. **LoA with Fully Accepted (04 April 2021)**
5. Proof Reading (11 April 2021)

Telkom University

SUYANTO SUYANTO <suyanto@telkomuniversity.ac.id>

Decision on submission to Journal of Computational Science

1 message

Journal of Computational Science <em@editorialmanager.com> Sun, Apr 4, 2021 at 6:08 PM
Reply-To: Journal of Computational Science <jocs@elsevier.com>
To: Suyanto Suyanto <suyanto@telkomuniversity.ac.id>

Manuscript Number: JOCSCI-D-21-00030R1
Evolutionary Rao Algorithm

Dear Dr. Suyanto,

Thank you for submitting your manuscript to Journal of Computational Science.

I am pleased to inform you that your manuscript has been accepted for publication.

My comments, and any reviewer comments, are below.
Your accepted manuscript will now be transferred to our production department. We will create a proof which you will be asked to check, and you will also be asked to complete a number of online forms required for publication. If we need additional information from you during the production process, we will contact you directly.

We appreciate you submitting your manuscript to Journal of Computational Science and hope you will consider us again for future submissions.

Kind regards,
Valeria Krzhizhanovskaya
Editor-in-Chief
Journal of Computational Science

==============
Editor and Reviewer comments:

Based on the reviewer's comments as given below, the authors addressed their requested issues and remarks as providing new results on benchmark functions (CEC-C06 2019) and real applications (Global Trajectory Optimization Problems from the European Space Agency) with comparing with Firefly Algorithm. Now, the manuscript can be accepted for publication.

Reviewer #1: In the new version of the manuscript, the authors provide new results on benchmark functions (CEC-C06 2019) and real applications (Global Trajectory Optimisation Problems from the European Space Agency). Moreover, they compare their approach with the Firefly Algorithm they draw inspiration from.

The authors addressed all my suggestions and remarks. I have no more comment.

Reviewer #2: The authors properly handled my comments.

Reviewer #3: The authors have addressed the reviewers concern. Therefore, the manuscript can be accepted.

===========

More information and support

FAQ: When and how will I receive the proofs of my article?
https://service.elsevier.com/app/answers/detail/a_id/6007/p/10592/supporthub/publishing/related/

You will find information relevant for you as an author on Elsevier’s Author Hub: https://www.elsevier.com/authors

FAQ: How can I reset a forgotten password?
https://service.elsevier.com/app/answers/detail/a_id/28452/supporthub/publishing/
For further assistance, please visit our customer service site: https://service.elsevier.com/app/home/supporthub/publishing/
Here you can search for solutions on a range of topics, find answers to frequently asked questions, and learn more about Editorial Manager via interactive tutorials. You can also talk 24/7 to our customer support team by phone and 24/7 by live chat and email

---

Evidence of correspondence

Evolutionary Rao Algorithm

1. First submission (04 January 2021)
2. LoA with Major Revision (23 February 2021)
3. Responses to Reviewers, Final submission (12 March 2021)
4. LoA with Fully Accepted (04 April 2021)
5. **Proof Reading (11 April 2021)**

ELSEVIER

Publishing Agreement

Elsevier B.V.

Evolutionary Rao Algorithm

Corresponding author Dr. Suyanto Suyanto
E-mail address suyanto@telkomuniversity.ac.id
Journal Journal of Computational Science
Article number 101368
Our reference JOCS_101368
PII S1877-7503(21)00059-4

Your Status

- I am one author signing on behalf of all co-authors of the manuscript

Assignment of Copyright

I hereby assign to Elsevier B.V. the copyright in the manuscript identified above (where Crown Copyright is asserted, authors agree to grant an exclusive publishing and distribution license) and any tables, illustrations or other material submitted for publication as part of the manuscript (the "Article"). This assignment of rights means that I have granted to Elsevier B.V., the exclusive right to publish and reproduce the Article, or any part of the Article, in print, electronic and all other media (whether now known or later developed), in any form, in all languages, throughout the world, for the full term of copyright, and the right to license others to do the same, effective when the Article is accepted for publication. This includes the right to enforce the rights granted hereunder against third parties.

Supplemental Materials

"Supplemental Materials" shall mean materials published as a supplemental part of the Article, including but not limited to graphical, illustrative, video and audio material.

With respect to any Supplemental Materials that I submit, Elsevier B.V. shall have a perpetual worldwide, non-exclusive right and license to publish, extract, reformat, adapt, build upon, index, redistribute, link to and otherwise use all or any part of the Supplemental Materials in all forms and media (whether now known or later developed), and to permit others to do so.

Research Data

"Research Data" shall mean the result of observations or experimentation that validate research findings and that are published separate to the Article, which can include but are not limited to raw data, processed data, software, algorithms, protocols, and methods.

With respect to any Research Data that I wish to make accessible on a site or through a service of Elsevier B.V., Elsevier B.V. shall have a perpetual worldwide, non-exclusive right and license to publish, extract, reformat, adapt, build upon, index, redistribute, link to and otherwise use all or any part of the Research Data in all forms and media (whether now known or later developed) and to permit others to do so. Where I have selected a specific end user license under which the Research Data is to be made available on a site or through a service, the publisher shall apply that end user license to the Research Data on that site or service.

Reversion of rights

Articles may sometimes be accepted for publication but later rejected in the publication process, even in some cases after public posting in "Articles in Press" form, in which case all rights will revert to the author (see https://www.elsevier.com/about/our-business/policies/article-withdrawal).

Revisions and Addenda

I understand that no revisions, additional terms or addenda to this Journal Publishing Agreement can be accepted without Elsevier B.V.'s express written consent. I understand that this Journal Publishing Agreement supersedes any previous agreements I have entered into with Elsevier B.V. in relation to the Article from the date hereof.

Author Rights for Scholarly Purposes

I understand that I retain or am hereby granted (without the need to obtain further permission) the Author Rights (see description below), and that no rights in patents, trademarks or other intellectual property rights are transferred to Elsevier B.V..
The Author Rights include the right to use the Preprint , Accepted Manuscript and the Published Journal Article for Personal Use and Internal Institutional Use . They also include the right to use these different versions of the Article for Scholarly Sharing purposes, which include sharing:

- the Preprint on any website or repository at any time;
- the Accepted Manuscript on certain websites and usually after an embargo period;
- the Published Journal Article only privately on certain websites, unless otherwise agreed by Elsevier B.V..

In the case of the Accepted Manuscript and the Published Journal Article the Author Rights exclude Commercial Use (unless expressly agreed in writing by Elsevier B.V.), other than use by the author in a subsequent compilation of the author's works or to extend the Article to book length form or re-use by the author of portions or excerpts in other works (with full acknowledgment of the original publication of the Article).

Author Representations / Ethics and Disclosure / Sanctions

I affirm the Author Representations noted below, and confirm that I have reviewed and complied with the relevant Instructions to Authors, Ethics in Publishing policy, Declarations of Interest disclosure and information for authors from countries affected by sanctions (Iran, Cuba, Sudan, Burma, Syria, or Crimea). Please note that some journals may require that all co-authors sign and submit Declarations of Interest disclosure forms. I am also aware of the publisher's policies with respect to retractions and withdrawal (https://www.elsevier.com/about/our-business/policies/article-withdrawal).

For further information see the publishing ethics page at https://www.elsevier.com/about/our-business/policies/publishing-ethics and the journal home page. For further information on sanctions, see https://www.elsevier.com/about/our-business/policies/trade-sanctions

Author representations

- The Article I have submitted to the journal for review is original, has been written by the stated authors and has not been previously published.
- The Article was not submitted for review to another journal while under review by this journal and will not be submitted to any other journal.
- The Article and the Supplemental Materials do not infringe any copyright, violate any other intellectual property, privacy or other rights of any person or entity, or contain any libellous or other unlawful matter.
- I have obtained written permission from copyright owners for any excerpts from copyrighted works that are included and have credited the sources in the Article or the Supplemental Materials.
- Except as expressly set out in this Journal Publishing Agreement, the Article is not subject to any prior rights or licenses and, if my or any of my co-authors' institution has a policy that might restrict my ability to grant the rights required by this Journal Publishing Agreement (taking into account the Author Rights permitted hereunder, including Internal Institutional Use), a written waiver of that policy has been obtained.
- If I and/or any of my co-authors reside in Iran, Cuba, Sudan, Burma, Syria, or Crimea, the Article has been prepared in a personal, academic or research capacity and not as an official representative or otherwise on behalf of the relevant government or institution.
- If I am using any personal details or images of patients, research subjects or other individuals, I have obtained all consents required by applicable law and complied with the publisher's policies relating to the use of such images or personal information. See https://www.elsevier.com/about/our-business/policies/patient-consent for further information.
- Any software contained in the Supplemental Materials is free from viruses, contaminants or worms.
- If the Article or any of the Supplemental Materials were prepared jointly with other authors, I have informed the co-author(s) of the terms of this Journal Publishing Agreement and that I am signing on their behalf as their agent, and I am authorized to do so.

Governing Law and Jurisdiction

This Agreement will be governed by and construed in accordance with the laws of the country or state of Elsevier B.V. ("the Governing State"), without regard to conflict of law principles, and the parties irrevocably consent to the exclusive jurisdiction of the courts of the Governing State.

For information on the publisher's copyright and access policies, please see http://www.elsevier.com/copyright.
For more information about the definitions relating to this agreement click here.

☑ **I have read and agree to the terms of the Journal Publishing Agreement.**

7 April 2021 T-copyright-v22/2017

Evolutionary Rao Aalgorithm★★

The corrections made in this section will be reviewed and approved by a journal production editor.

Q1 Suyanto Suyanto* suyanto@telkomuniversity.ac.id, Agung Toto Wibowo agungtoto@telkomuniversity.ac.id, Said Al Faraby saidalfaraby@telkomuniversity.ac.id, Siti Saadah sitisaadah@telkomuniversity.ac.id, Rita Rismala ritaris@telkomuniversity.ac.id

School of Computing, Telkom University, Bandung, Indonesia

*Corresponding author.

Abstract

This paper proposes an evolutionary Rao algorithm (ERA) to enhance three state-of-the-art metaheuristic Rao algorithms (Rao-1, Rao-2, Rao-3) by introducing two new schemes. Firstly, the population is split into two sub-populations based on their qualities: high and low, with a particular portion. The high-quality sub-population searches for an optimum solution in an exploitative manner using a movement scheme used in the Rao-3 algorithm. Meanwhile, the low-quality one does in an explorative fashion using a new random walk. Secondly, two evolutionary operators: crossover and mutation, are incorporated to provide both exploitation and exploration strategies. A fitness-based adaptation is introduced to dynamically tune the three parameters: the portion of high-quality individuals, mutation radius, and mutation rate throughout the evolution, based on the improvement of best-so-far fitness. In contrast, the crossover is implemented using a standard random scheme. Comprehensive examinations using 38 benchmarks: twenty-three classic functions, ten CEC-C06 2019 benchmarks, and five global trajectory optimization problems show that the proposed ERA generally outperforms the four competitors: Rao-1, Rao-2, Rao-3, and firefly algorithm with courtship learning (FA-CL). Detailed investigations indicate that both proposed schemes work very well to make ERA evolves in an exploitative manner, which is created by a high portion of high-quality individuals and the crossover operator, and avoids being trapped on the local optimum solutions in an explorative manner, which is generated by a high portion of low-quality individuals and the mutation operator. Finally, the adaptation scheme effectively controls the exploitation-exploration balance by dynamically tuning the portion, mutation radius, and mutation rate throughout the evolution process.

Keywords: Evolutionary Rao algorithm; Exploitation-exploration balance; Fitness-based adaptation scheme; Random walk; Two subpopulations

1 Introduction

The metaheuristic optimization algorithms [Instruction: "... that can be categorized ..." --> "... can be categorized ..."]that can be categorized into two groups: evolutionary algorithms (EAs) and swarm intelligence (SI) algorithms [1]. EAs are inspired by both evolution and natural selection, such as Genetic Algorithm (GA) [2], [3], Evolution Strategies (ES) [4], [5], and Differential Evolution (DE) [6]. Meanwhile, SI algorithms are inspired by a natural swarm, such as Particle Swarm Optimization (PSO) [7], [8], Firefly Algorithm (FA) [9], [10], Grey Wolf Optimizer (GWO) [11], [12], and Ant Lion Optimization (ALO) [13].

GA is one of the most popular EAs introduced in the 1970s [14]. It uses both evolution and natural selection that are applied to its population over generations. A population consists of some individual chromosomes, each representing a candidate solution. The new chromosomes in a generation are either some of the best chromosomes (elitism) in the previous generation or generated by genetic operations, such as crossover and mutation. The crossover takes two chromosomes and produces one offspring inherited part of chromosome values from each parent. In contrast, the mutation is randomly changing some values in a chromosome. The mutation is responsible for exploration, while crossover and elitism direct toward exploitation. GA can avoid being trapped in the local optima. It is also applicable to non-differentiable and high dimensionality functions. On the other hand, it converges slowly because of the highly-random operations that do not give a clear direction to find the global optimum solution quickly. However, various improvement schemes have been proposed to overcome the drawback, such as a concept of human-like constrained-mating [15] that creates a more explorative search strategy.

In 1995, the Particle Swarm Optimisation (PSO) was introduced by Kennedy and Elberhart [16]. The movements of the particles in searching for a global optimum mimics the behavior of bird flocking and fish schooling. PSO is one of the most popular SI algorithms since it has three advantages: easy to implement, few parameters that are simply tuned, and effective in searching the global optimum solution since it has a clearer direction than GA. However, it tends to prematurely converge on a local optimum in optimizing a multimodal function since it uses a static finite leader and group based on a linear movement. Therefore, some strategies are developed to tackle the issue, such as a learning structure [17] to decouple exploration and exploitation and a dynamic updating of the inertia weights [18] to control the convergence.

In 2009, the Firefly Algorithm (FA) was proposed [19]. In FA, each firefly will be attracted to all other brighter (better) fireflies, not only to the global best like in PSO. Also, the brighter firefly's attractiveness is decreased proportioned to the distance between the two fireflies due to the light absorption. Since the fireflies will usually be attracted more to their brighter neighbor than the further away brightest individual, the exploration is more effective than PSO. In other words, FA uses a dynamic leader and group based on a nonlinear movement. Moreover, FA can be turned into PSO by setting the light absorption parameter such that every firefly can be seen clearly by all other fireflies. Consequently, all fireflies will be attracted to the brightest one (global best). In some experiments, FA shows better performance than PSO due to two critical characteristics [20]: 1) FA usually divides its population into a subgroup, 2) By not having an explicit global best, FA can avoid premature convergence. Several improved schemes are created to enhance the FA performance, such as a courtship learning framework [21], where the population is divided into subpopulations: female and male, to improve the convergence speed and solution accuracy. Another improvement scheme is the best neighbor guided strategy [22], where each firefly is attracted to the best firefly among some randomly chosen neighbors to decrease the firefly oscillations in every attraction-induced migration stage as well as increase the probability of the guidance a new better direction.

In 2014, Grey Wolf Optimization (GWO) was introduced by Mirjalili [23]. It is inspired by both the social hierarchy and hunting methods of grey wolves (GWs). The hierarchy of GWs has four groups: alpha, beta, delta, and omegas. GWO selects the three fittest wolves (best solutions) as the alpha, beta, and delta, while the rest as omegas. The hunting process of GWs is guided by the three fittest wolves. All omegas follow them. It has four phases, which are mathematically modeled into four behaviors: Harassing Prey, Hunting, Attacking, and Searching, that create a high exploitative searching strategy. It quickly converges to an optimum solution for unimodal functions. However, it suffers from multimodal functions since it has a low explorative movement. Therefore, some variants of GWO are developed by incorporating various mechanisms/operators, such as differential evolution with elimination mechanism [24], simulated annealing [25], or refraction learning operator [26]. GWO can also be improved using a dimension learning-based hunting movement strategy [27], which uses a different approach to construct a neighborhood for each wolf to enhance the balance of local and global searches and maintain diversity.

In 2015, Ant Lion Optimizer (ALO) was proposed by Mirjalili [28]. ALO mimics the interaction between antlions and ants in the trap, where ants move over the search space and antlions hunt them and become fitter using traps. A new random walk is introduced to model the ant's movement as they move stochastically in nature to find some food. It has high exploitation and convergence speed because of the adaptive boundary shrinking mechanism and elitism. It also high exploration due to the random walk and roulette wheel selection mechanisms. However, although it has few parameters, some schemes and movements make ALO seems too-complicated. Hence, some versions of ALO are created by modifying, hybridizing, and providing an ability to solve a multi-objective problem [13].

In 2020, the metaphor-less optimization methods called Rao algorithms were proposed by Ravipudi Venkata Rao [29]. The Rao Algorithms use both best and worst solutions in each iteration and the random interactions among the candidate solutions to quickly find an optimum solution. They need two standard parameters: population size and a maximum number of evaluations that easy to adjust. They drop many parameters used in the previous metaphor-based algorithms, such as cohesion, intensity, probability, and other commonly challenging parameters to tune carefully.

The Rao algorithms have three variants: Rao-1, Rao-2, and Rao-3, which respectively use three different equations below:

$$X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,best,i} - X_{j,worst,i}) \tag{1}$$

$$X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,best,i} - X_{j,worst,i}) + r_{2,j,i}(|X_{j,k,i} \text{ or } X_{j,l,i}| - |X_{j,l,i} \text{ or } X_{j,k,i}|), \tag{2}$$

$$X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,best,i} - |X_{j,worst,i}|) + r_{2,j,i}(|X_{j,k,i} \text{ or } X_{j,l,i}| - (X_{j,l,i} \text{ or } X_{j,k,i})), \tag{3}$$

where $X_{j,best,i}$ represents the best candidate as the value of variable j, and $X_{j,worst,i}$ represents the worst candidate as value of variable j, both throughout the i-th iteration. $X'_{j,k,i}$ is the updated value after the equation, and both $r_{1,j,i}$ as well as $r_{2,j,i}$ are randomly generated in [0,1] for the j~~-th variable throughout the i-~~th variable throughout the ith iteration. In the term $|X_{j,k,i} \text{ or } X_{j,l,i}|$, the candidate solution k is compared to another candidate l, which is randomly selected from the available candidates in the population. The term $|X_{j,k,i}|$ is selected if k is fitter than l. Otherwise, the $|X_{j,l,i}|$ is chosen. The same rule is applied to [Instruction: "... the second the term ..." --> "... the second term ..."]the second the term $(X_{j,l,i} \text{ or } X_{j,k,i})$.

All formulas used in the three Rao algorithms are similar to GWO, making them more exploitative than explorative. Using both best and worst solutions, they converge to an optimum solution for unimodal functions more quickly than GWO. However, with low explorative movement, they can be worse for multimodal functions. As described in [29], Rao is easy to get stuck in multimodal functions. Rao-3 gives a better solution in the Schwefel function from the six benchmark multimodal-functions and much worse for the other five benchmark multimodal-functions.

Therefore, in this research, an evolutionary Rao algorithm (ERA) is proposed to enhance the three original Rao algorithms by introducing two additional schemes. Firstly, the population is split into two sub-populations based on their qualities: high and low, with a particular portion depending on the given problem. The high-quality sub-population searches for an optimum solution in an exploitative manner using a movement scheme used in the Rao-3 algorithm. Meanwhile, the low-quality one does in an explorative fashion using a new random walk introduced in this research. This scheme is similar to the courtship learning framework in the Enhanced FA [21], where the population is also divided into two subpopulations: female and male, but ERA uses a predefined specific portion. Secondly, two evolutionary operators: crossover and mutation, are used to give exploitation and exploration searching strategies. A fitness-based adaptation is introduced to dynamically tune the ~~the~~ portion of high-quality individuals, mutation radius, and mutation rate during the evolution. Meanwhile, the crossover is implemented using a random scheme with the common probabilistic values that do not create any additional parameters. The ERA is finally examined and compared to the three original Rao algorithms [29] as well as the firefly algorithm with courtship learning (FA-CL) [21] using three groups of benchmark functions: 1) the classic benchmark functions that contain seven unimodal, six multimodal, and ten low-dimension multimodal; 2) the CEC-C06 2019 test suites that consists of ten benchmark functions [30]; and 3) the global trajectory optimization problems provided by European Space Agency that contains five real problems of Cassini1, GTOC1, Messenger, Sagas, and Cassini2 [31].

2 Proposed ~~Evolutionary Rao A~~evolutionary Rao algorithm

The pseudo-code of ERA is illustrated in Algorithm 1. In the initial phase, define the fixed population size p, the initial portion of high-quality (HQ) individuals $s = 0.5$, the initial mutation radius $a = 0.5$, the initial mutation rate $b = 0.9$, and randomly initialize the population of p individuals. In the next phase, the evolution is performed until a stopping condition is reached, such as when the number of evaluations is equal to the given maximum limit.

In each generation, six steps are carried out. Firstly, the quality of each individual is calculated; and their quality-ranks are then sorted in the descending mode. Secondly, the population is split into two sub-populations: high-quality (HQ) and low-quality (LQ), with the defined portion s, and both the best individual X_{best} and the worst individual X_{worst} are selected. Thirdly, each HQ individual is moved to follow the X_{best} using Eq. (3). Fourthly, the fittest HQ individual is selected as the BestHQ, and then one of the two evolutionary operators is chosen: crossover (exploitative) or mutation (explorative), to move the X_{best}. Fifthly, each LQ individual is moved using a new random walk. Finally, the fitness-based adaptation is performed by updating s, a, and b based on the improvement or stagnation of two consecutive best-so-far fitness.

Algorithm 1 Evolutionary Rao ~~A~~algorithm

Result: X_{best} as the optimum solution
Set p as the fixed population size (number of individuals);
Set $s = 0.5$, $a = 0.5$, and $b = 0.9$ as the initial values of high-quality (HQ) individuals portion, mutation radius, and mutation rate, respectively;
Randomly initialize the population of p individuals;
while *StoppingCondition* $=$ *false* **do**
  for each individual, calculate its quality and then sort the quality-ranks in the descending mode;
  Select the fittest individual as the X_{best};
  Select the most fit individuals with the defined portion s as the HQ and the rests as the low-quality (LQ) individuals;
  Select the lowest-quality individual as the X_{worst};
  for each HQ individual, move it to follow the X_{best} using Eq. 3;
  Select the fittest HQ individual as the BestHQ;
  if *rand* $>$ *0.5* **then**
    Offsprings = Crossover(BestHQ, X_{best});
    Replacement(BestHQ, X_{best}, Offsprings);
  else
    Offspring = Mutation(X_{best});
    Replacement(X_{best}, Offspring);
  end
  for each LQ individual move it to follow or distract a randomly selected HQ individual on the half of dimensions using Eq. (9);
  if *two consecutive best-so-far fitness show an improvement* **then**
    Increase s, but decrease a and b, using Eq. (10), (11), and (12);
  else
    Decrease s, but increase a and b, using Eq. (10), (11), and (12);
    Mutate $(1 - s) \times p$ low-quality individuals;
  end
end

2.1 Two sub-populations

The population of p candidate solutions (individuals) is split into two sub-populations based on their qualities: high and low, with a proper portion based on the given problem. The high-quality (HQ) sub-population searches for an optimum solution in an exploitative manner using the same movement scheme as in the Rao-3 algorithm. Meanwhile, the low-quality (LQ) one does in an explorative fashion using a new random walk introduced in this research. Hence, this scheme creates a new parameter s: the portion of high and low-quality individuals in the population. It is in the interval

(0, 1) and easy to adjust. Hypothetically, it should be high (more than 0.5) to make ERA more exploitative and faster to optimize the unimodal functions. In contrast, it must be low (less than 0.5) to make ERA more explorative to solve the multimodal functions. A fitness-based adaptation scheme is proposed to increase or decrease the portion s automatically based on the best-so-far fitness during the evolution. If two consecutive best-so-far fitness values show an improvement, then the portion s is decreased to make ERA more exploitative. In contrast, if two consecutive best-so-far fitness shows a stagnation, then the portion s is increased to make ERA more explorative. A detailed explanation will be provided in Section 2.5.

Furthermore, the population of p individuals is split into two subpopulations: the high-quality subpopulation of h individuals and the low-quality sub-population of l individuals, which are calculated as

$$h = \lfloor (p-1) \times s \rfloor, \tag{4}$$

$$l = (p-1) - h, \tag{5}$$

where s is the portion of HQ individuals in the population. However, both Eq~~. (4) and Eq.~~s. (4) and (5) may produce zero for either h or l if the portion s is too-small or too-high. Hence, an enforcement procedure is implemented to ensure that a too-small s makes the HQ sub-population consists of at least two individuals, and a too-big s also makes the LQ sub-population contains at least two individuals.

2.2 Crossover

The crossover is implemented using a whole arithmetic crossover, which is defined as

$$\begin{aligned} X' &= r \cdot X + (1-r) \cdot Y \\ Y' &= r \cdot Y + (1-r) \cdot X \end{aligned} \tag{6}$$

where r is a randomly generated number in the interval (0, 1), which should be not equal to 0.5 to prevent generating the same two offsprings (new individuals); if $r = 0.5$, then both offsprings X' and Y' are the same as the average of both current individuals X and Y. Hence, this crossover scheme does not need any user parameter.

2.3 Mutation

The mutation is simply implemented using a creep mutation by adding a small value (positive or negative) to each mutated element. The small value is randomly generated using a Gaussian probability that is symmetric, distributed on 0, and has a high probability for the smaller values. The creep mutation is defined as

$$\langle x_1, x_2, \ldots, x_n \rangle \rightarrow \langle x'_1, x'_2, \ldots, x'_n \rangle, \tag{7}$$

$$x'_i = \begin{cases} x_i + (2r_1 - 1) \times a |U_i - L_i|, & \text{if } r_2 < b \\ x_i, & \text{otherwise,} \end{cases} \tag{8}$$

where $x_1, x_2, \ldots x_n \in [L_i, U_i]$, L_i and U_i are the lower and upper bounds of the interval of the ith element (variable or dimension), r_1 and r_2 are random values with the normal distribution in the interval [0, 1], and a and b are the mutation radius and the mutation rate, respectively, which are automatically tuned using a fitness-based adaptation scheme that will be described in Section 2.5.

2.4 Random walk

To provide an ability to search for an optimum solution in an explorative manner, each LQ individual is moved using a new random walk formulated as

$$X'_{m,LQ,i} = X_{m,LQ,i} + r_{1,m,i}(X_{m,HQ,n} - X_{m,LQ,i}) \tag{9}$$

where $X_{m,LQ,i}$ and $X_{m,HQ,n}$ is the LQ individual i and the HQ individual n (randomly selected from the high-quality sub-population), respectively, and m is the randomly selected dimension; not all dimensions are used here to make this random walk more explorative.

2.5 Fitness-based adaptation scheme

Based on the above description, ERA has four parameters: population size p, portion s, mutation-radius a, and mutation-rate b. Hypothetically, p is the most robust parameter. In contrast, s, a, and b are estimated quite sensitive since they control the exploration strategy. Therefore, these three parameters are designed to be tuned adaptively during the evolution. A new simple fitness-based adaptation scheme based on the fitness values of the best-so-far individual is proposed for this purpose. If two consecutive best-so-far fitness values show an improvement, then s is increased, but both a and b are decreased, to make ERA more exploitative. In contrast, if two consecutive best-so-far fitness shows a stagnation, then s is decreased, both a and b are increased to make ERA more explorative, and all low-quality individuals are mutated using both new a and b to spread them in new locations. The increment and decrement are formulated as follow:

$$s' = \begin{cases} s \times (1 - \frac{\Delta f_1 + \Delta f_2}{2}), \text{ if } \Delta f_1 > 0 \text{ and } \Delta f_2 > 0 \\ s \times 0.97, \text{ if } \Delta f_1 = 0 \text{ and } \Delta f_2 = 0 \end{cases} \tag{10}$$

$$a' = \begin{cases} a \times 0.97, \text{ if } \Delta f_1 > 0 \text{ and } \Delta f_2 > 0 \\ a \times 1.03, \text{ if } \Delta f_1 = 0 \text{ and } \Delta f_2 = 0 \end{cases} \tag{11}$$

$$b' = \begin{cases} b \times 0.97, \text{ if } \Delta f_1 > 0 \text{ and } \Delta f_2 > 0 \\ b \times 1.03, \text{ if } \Delta f_1 = 0 \text{ and } \Delta f_2 = 0 \end{cases} \tag{12}$$

where $\Delta f_1 = \frac{|f_1 - f_2|}{f_1}$ and $\Delta f_2 = \frac{|f_2 - f_3|}{f_2}$ are the first and the second differences of the fitness values of two consecutive generations during the evolution process, respectively.

Moreover, the initial, minimum, and maximum values for those three parameters can be easily defined. Since the characteristics of the given problem are unknown, then the initial portion s is set as 0.5, while the minimum and the maximum values are set to 0.1 and 0.9, respectively. Next, both minimum and maximum values of a are set as 0.05 and 0.5, respectively. It means the mutation of an element (dimension) can occur in the radius of 5~~to 50~~% to 50% out of the search space. In other words, an individual can be mutated at the maximum range of [-0.5, 0.5] in the search space. Hence, the mutation can cover the whole search space. Next, the initial value of a is tuned as 0.5 to provide the maximum exploration in the beginning iterations of the evolution process. Finally, b is defined in the interval [0.1, 0.9], and its initial value is 0.9 to maximize the exploration strategy in the beginning evolution process. Using the maximum mutation radius and rate, ERA can have a high-exploration ability to handle the effects of shift and rotation of the test functions, such as in the CEC-C06 2019 benchmark functions.

2.6 Complexity analysis of ERA

The mathematical complexity of ERA can be analyzed as follows. For each iteration, ERA has a time complexity of $O(p \times n + p \times c + \log p)$, where p is the population size, n is the dimension of the given problem, c is the complexity of the objective function calculation, and $\log p$ is the complexity of the fitness sorting to split the population into HQ and LQ sub-populations. It is clear that compared to the original Rao, ERA is slightly more complicated because of the additional sorting complexity of $\log p$. Meanwhile, the complexity of the fitness-based adaptation scheme can be ignored since it is quite low; it only contains addition, substraction, and logical operations.

3 Results and ~~D~~discussion

In this research, twenty-three benchmark functions: seven unimodal, six multimodal, and ten low-dimension multimodal functions [29] are used to investigate both exploitation and exploration abilities of the proposed ERA. Table 1 illustrates the benchmark functions with their identities (ID), names, types, dimensions, ranges, and global optimum values $f_{\min}$. Meanwhile, their two-dimensional views are illustrated in [Instruction: Please update "Figure 1" --> "Fig. 1" to make it consistent with all the others.]Figure 1. Seven benchmark functions, with ID = 1 to 7, are unimodal to examine the exploitation ability. Next, six benchmark functions, ID = 8 to 13, are multimodal, with many local optima increasing as the dimension increases, to evaluate the exploration ability. Finally, ten functions, ID = 14 to 23, are low-dimension multimodal (LDM) to investigate the exploration ability in the case of low-dimension optimization problems.

Table 1

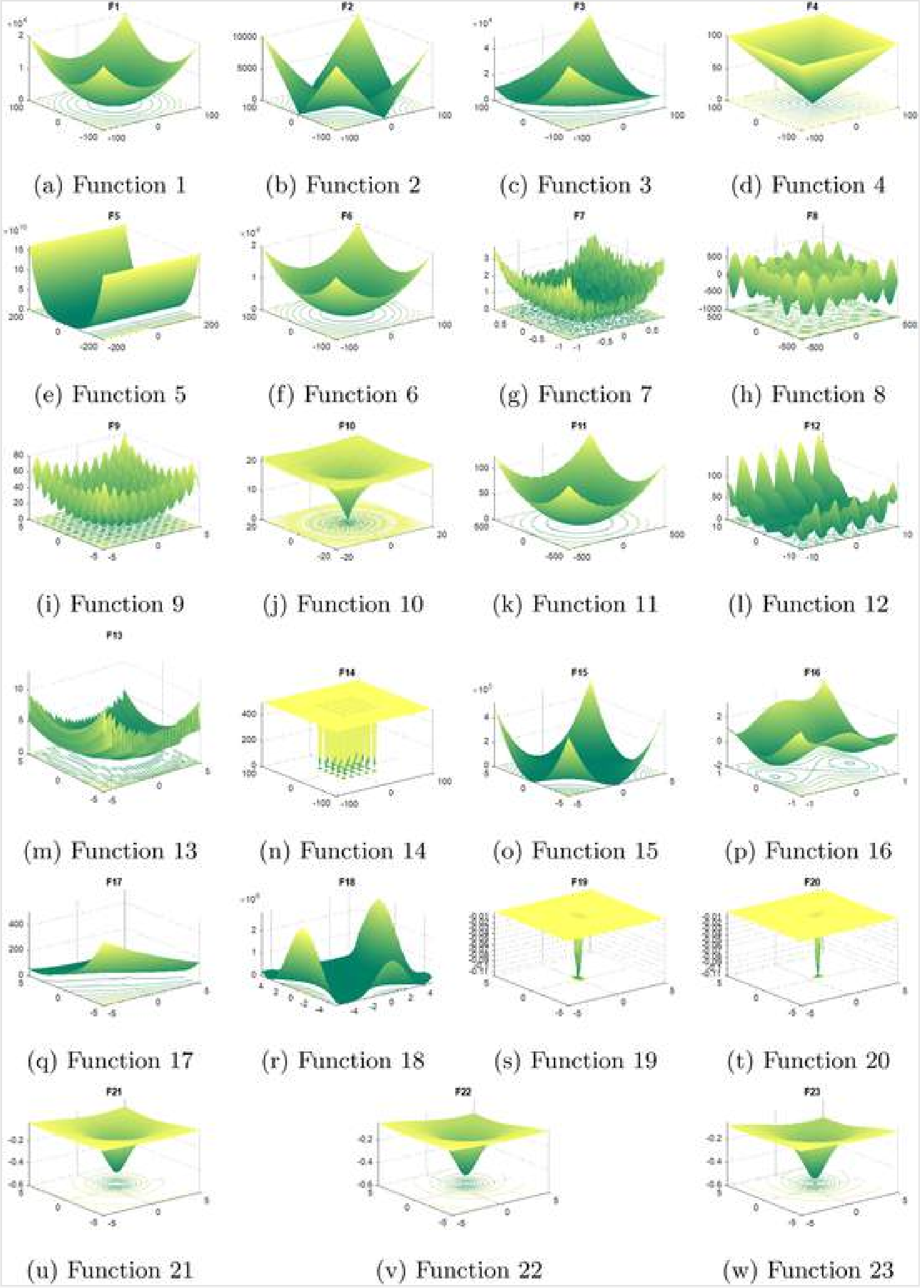
The table layout displayed in this section is not how it will appear in the final version. The representation below is solely purposed for providing corrections to the table. To preview the actual presentation of the table, please view the Proof.

Twenty three classic benchmark functions

Func	Name	Type	Dim	Range	$f_{\min}$
CF1	Sphere	Unimodal	30	[−-100, 100]	0
CF2	Schwefel 2.22	Unimodal	30	[−-100, 100]	0
CF3	Schwefel 1.2	Unimodal	30	[−-100, 100]	0
CF4	Schwefel 2.21	Unimodal	30	[−-100, 100]	0
CF5	Rosenbrock	Unimodal	30	[−-30, 30]	0
CF6	Step	Unimodal	30	[−-100, 100]	0
CF7	Quartic	Unimodal	30	[−-1.28, 1.28]	0
CF8	Schwefel	Multimodal	30	[−~~500, 500]~~500, 500]	−418.9829 × Dim
CF9	Rastrigin	Multimodal	30	[−-5.12, 5.12]	0
CF10	Ackley	Multimodal	30	[−-32, 32]	0
CF11	Griewank	Multimodal	30	[−-600, 600]	0
CF12	Penalized	Multimodal	30	[−-50, 50]	0
CF13	Penalized2	Multimodal	30	[−-50, 50]	0
CF14	Foxholes	LDM	2	[−-65, 65]	0.998
CF15	Kowalik	LDM	4	[−-5, 5]	0.0003
CF16	Six Hump Camel	LDM	2	[−~~5, 5]~~5, 5]	−1.0316
CF17	Branin	LDM	2	[−-5, 5]	0.398
CF18	GoldStein-Price	LDM	2	[−-2, 2]	3
CF19	Hartman 3	LDM	3	[0, 1]	−-3.86

CF20	Hartman 6	LDM	6	[0, 1]	−-3.32
CF21	Shekel 5	LDM	4	[0, 10]	−-10.1532
CF22	Shekel 7	LDM	4	[0, 10]	−-10.4029
CF23	Shekel 10	LDM	4	[0, 10]	−-10.5364

Fig. 1



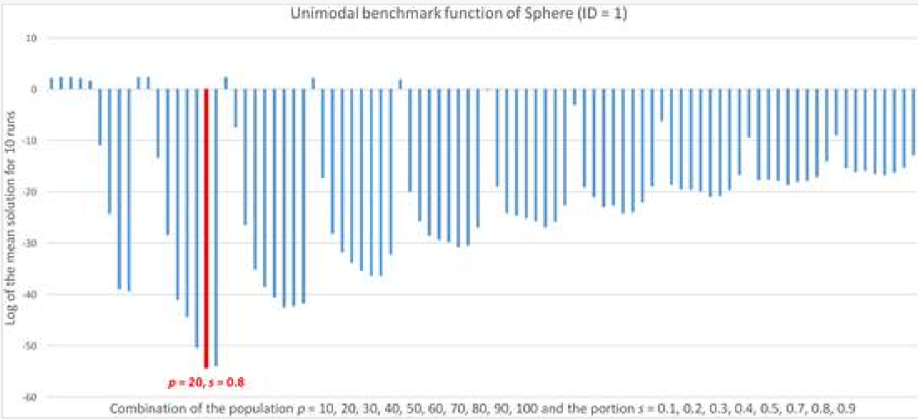
Twenty three classic benchmark functions CF1 to CF23.

3.1 Preliminary observations

First, two parameters of ERA: population p and portion s, are observed to see their behaviors in optimizing the twenty-three classic benchmark functions. For each function, ninety experiments are performed using combination of ~~ten~~10 values of p = 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 and nine values of s = 0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 0.8, 0.9, which can be defined as pairs of (10, 0.1), (10, 0.2),..., (100, 0.9). For each experiment, the maximum number of function evaluations is set to 30,000 with ten runs to reduce the coincidence. Here, only three experimental results of the representative benchmark functions are shown and discussed, namely unimodal (Sphere, ID = 1), multimodal (Schwefel, ID = 8), and low-dimension multimodal (Shekel 7, ID = 22), to see the behaviors of both parameters p and s in optimizing those three types of benchmark functions. The common parameter value of p is finally selected as a fixed-optimum value for all the benchmark functions. Meanwhile, the portion s is dynamically updated during the evolution process using a fitness-based adaptation scheme.

Fig. 2 illustrates the experimental results for the problem of searching a minimum solution to a unimodal function of Sphere (ID = 1), where the vertical axis uses $\log(\text{meansolution})$ to ensure the bar chart clearly shows all results from the ninety experiments. It can be seen that a too-small (10) or a big population p (30~~to~~–100) makes the ERA produces a bad solution. The bigger the p, the worse the solution. A small portion s (0.5 or less) also yields a poor solution. The smaller the s, the worse the solution. Hence, the combination of a too-big p and a too-small s is not recommended. The optimum combination is reached on $p = 20$ and $s = 0.8$. This result proves that a big portion of high-quality individuals in the small population makes the proposed ERA more exploitative and faster to find the optimum solution.

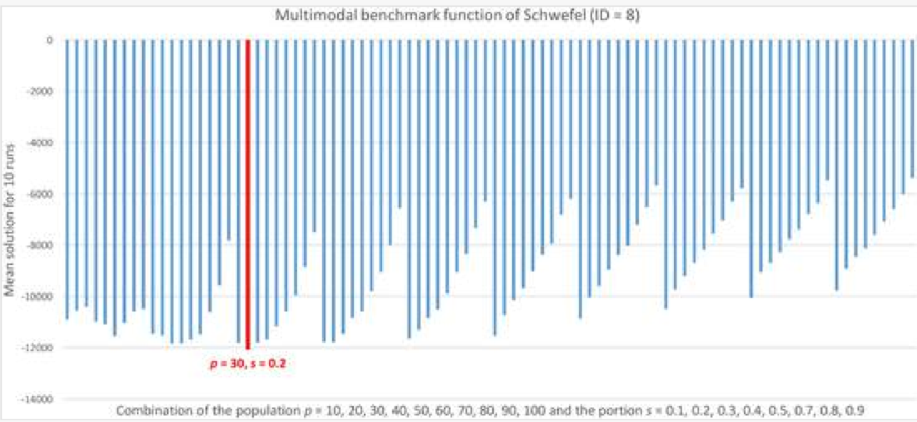
Fig. 2



Parameter tuning for a unimodal benchmark function of Sphere (ID = 1).

Next, Fig. 3 illustrates the ninety experimental results for the problem of minimizing a multimodal function of Schwefel (ID = 8). It informs that the portion s is sensitive, but the population size p is not; the bigger the s, the worse the solution. A too-big portion s drastically reduces the solution quality. The optimum combination is reached on $p = 30$ and $s = 0.2$. This result proves that a small portion of high-quality individuals in the small population makes the proposed ERA more explorative and faster to find the optimum solution to the multimodal functions with many local optima.

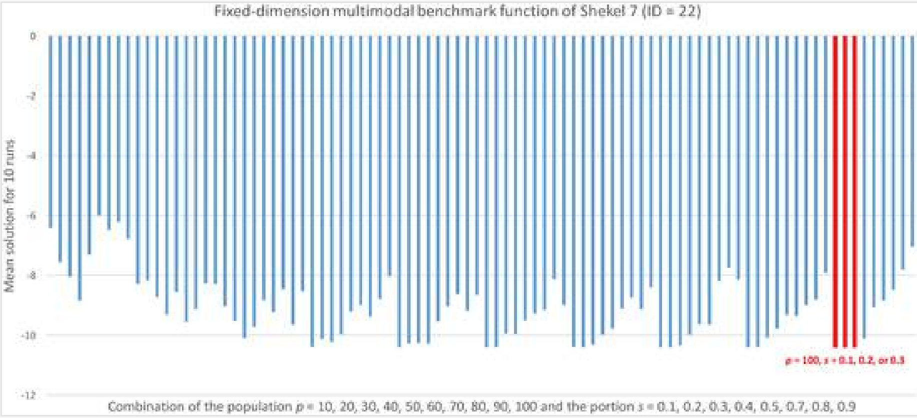
Fig. 3



Parameter tuning for a multimodal benchmark function of Schwefel (ID = 8).

Finally, Fig. 4 illustrates the ninety experimental results for the problem of minimizing a low-dimension multimodal function of Shekel 7 (ID = 22). It also informs that the portion s is sensitive, but the population size p is not; the bigger the s, the worse the solution. A too-big portion s drastically reduces the solution quality. The optimum combination is reached on a big $p = 100$ and a low $s = 0.2$. However, a smaller p up to 20 or 30 also gives a good solution. This result informs that a small portion of high-quality individuals in the big population makes ERA more explorative. Hence, it can search for an optimum solution to the low-dimension multimodal functions with a wide flat area.

Fig. 4



Parameter tuning for a low-dimension multimodal benchmark function of Shekel 7 (ID = 22)~~Fig. 5Friedman mean rank calculated using ten different population sizes p for each algorithm~~.

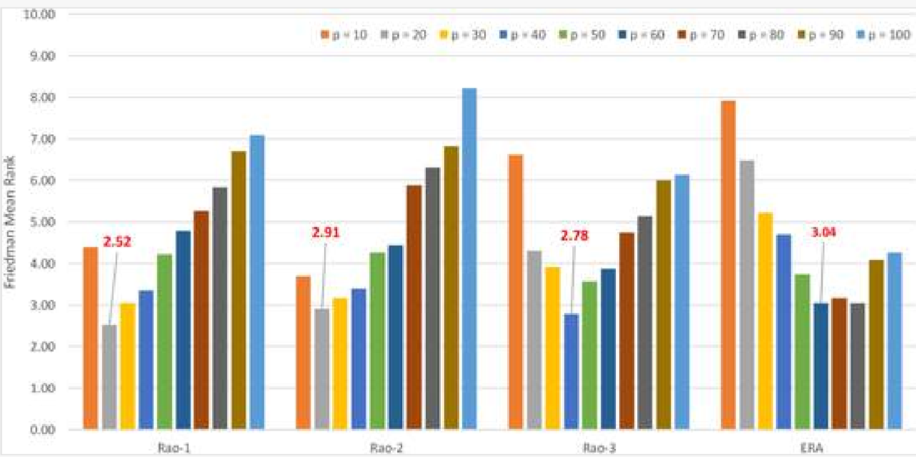
The three observations above prove the hypothesis that p is more robust than s. Therefore, the adaptation scheme is applied on s instead of p. A fitness-based adaptation of population size introduced in [32] is reported can improve the performance of the differential evolution, but that scheme is not used here since it will increase the complexity of ERA. Thus, p is designed to be a fixed value and tuned manually by doing a few experiments.

3.2 Parameter settings

Based on the research in [21], the best population size for FA-CL is 20. Thus, the parameter setting is focused on Rao-1, Rao-2, Rao-3, and ERA. Here, ten experiments with $p = 10, 20, ..., 100$ are carried out to find the optimum p for each algorithm based on the Friedman Mean Rank (FMR).

Fig. 5 illustrates the experimental results. The behavior of p is similar for Rao-1 and Rao-2. The smaller the p, the better the rank. The optimum value is reached on $p = 20$ for both algorithms. Meanwhile, p gives a different effect for Rao-3 that achieves the optimum value on $p = 40$. It also shows the different impacts for ERA, which gets the optimum value on $p = 60$. Finally, the parameter settings for ERA and other algorithms are listed in Table 2.

Fig. 5



Friedman mean rank calculated using 10 different population sizes p for each algorithm.

Table 2

The table layout displayed in this section is not how it will appear in the final version. The representation below is solely purposed for providing corrections to the table. To preview the actual presentation of the table, please view the Proof.

Parameter settings.

Algorithm	Parameter settings
Rao-1	$p = 20$
Rao-2	$p = 20$
Rao-3	$p = 40$
FA-CL	$p = 20, \alpha = 0.5, \beta_{\min} = 0.2, \beta = 1, \gamma = 1$
ERA	$p = 60, s = 0.5, a = 0.5, b = 0.9$

3.3 Evaluation on classic benchmark functions

First, the proposed ERA [Instruction: "is then examined" --> "is examined"]is then examined and compared with four other algorithms: Rao-1, Rao-2, Rao-3, and FA-CL to search the minimum solutions to the twenty-three benchmark functions listed in Table 1. For each benchmark function, the maximum number of function evaluations is set to 30,000 with 30 runs to reduce the coincidence. The random seeds of the 30 initial populations (for each benchmark function) are the same when the algorithms use the same population size p to get fairness. Otherwise, they are different. The Matlab source-codes used in the Rao-1, Rao-2 and Rao-3 refer to [29] while the one used in FA-CL refers to [21]. Meanwhile, the optimum parameter settings for all algorithms are described in Section 3.2. Table 3 illustrates the examination results based on five metrics (Met): best solution, worst solution, mean solution, standard deviation (STD), and mean function evaluations (MFE).

Table 3

The table layout displayed in this section is not how it will appear in the final version. The representation below is solely purposed for providing corrections to the table. To preview the actual presentation of the table, please view the Proof.

Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for 23 classic benchmark functions~~IDMetricRao-1Rao-2Rao-3FA-CLERA~~.

ID	Metric	Rao-1	Rao-2	Rao-3	FA-CL	ERA
1	Best	1.44026E-13	0.000206616	1.37198E-16	3360.332345	1.44407E-11
	Worst	1.56215E-11	0.05322429	3.57068E-13	7391.524099	7.07827E-10
	Mean	1.6427E-12	0.007910073	2.93612E-14	5494.642553	2.03341E-10
	STD	2.94821E-12	0.012500245	6.93888E-14	897.6329345	1.97333E-10
	MFE	30000	30000	30000	30176.8	30032
2	Best	9.88458E-08	0.046106637	3.48002E-09	20.78017429	2.1222E-06
	Worst	2.84524E-05	81.77760505	1.96246E-07	36.38814153	2.38276E-05
	Mean	1.75865E-06	6.122312306	4.70062E-08	31.53033206	8.63966E-06
	STD	5.11921E-06	16.17762336	5.36523E-08	3.444133931	4.73519E-06
	MFE	30000	30000	30000	30182.46667	30028.2
3	Best	29.58384537	24308.67108	5351.839864	6485.004266	983.5012373

	Worst	410.0667787	45693.37909	19092.21252	14405.71624	3470.902824
	Mean	149.4341504	35855.17634	11301.74245	10486.02575	2303.225697
	STD	105.4899095	5718.824871	3745.250011	1944.35227	728.8781476
	MFE	30000	30000	30000	30208.9	30023.6
4	Best	1.201443048	7.137837854	0.042098173	21.36789617	0.042019247
	Worst	12.36146637	28.08331631	59.65566877	36.06694776	0.300012918
	Mean	5.214910493	16.15411285	7.347592214	28.9938981	0.124708246
	STD	3.489736613	4.841469952	13.05854573	3.59291026	0.061288909
	MFE	30000	30000	30000	30226.43333	30025.6
5	Best	0.287292008	0.439075353	12.58554648	226704.144	18.30326575
	Worst	93.46438644	3019.406575	100.1684583	2564425.907	109.1279756
	Mean	35.67946414	130.9427468	35.87671438	1261919.835	31.40688996
	STD	29.6735389	546.4756884	27.26289807	584515.0346	17.69375467
	MFE	30000	30000	30000	30203.33333	30029.4
6	Best	2	0	0	3792	0
	Worst	53	12	3	7639	2
	Mean	10.2	1.833333333	0.3	5555.266667	0.2
	STD	9.219170432	3.006697505	0.651258728	1099.28108	0.484234198
	MFE	30000	28073.33333	13776	30144.96667	12368
7	Best	0.03452811	0.03404685	0.005036199	1.530176956	0.004682426
	Worst	0.211389877	0.168091645	0.081089247	4.106762973	0.036293801
	Mean	0.080890625	0.093085738	0.019933199	2.624059197	0.013211183
	STD	0.036979081	0.031284187	0.016182882	0.659081008	0.007750259
	MFE	30000	30000	30000	30145.5	30027.2
8	Best	– ~~10682.09946 10239.35633 11345.58004 4377.087113~~ 10682.09946	– 10239.35633	– 11345.58004	– 4377.087113	– 8879.935043
	Worst	– ~~3893.432932 5125.91502 3869.781006 3291.297315~~ 3893.432932	– 5125.91502	– 3869.781006	– 3291.297315	– 6882.950077
	Mean	– ~~6470.266849 8027.033295 8325.725086 3701.706264~~ 6470.266849	– 8027.033295	– 8325.725086	– 3701.706264	– 7753.907755
	STD	2090.801691	1321.261016	2361.956999	288.5432924	413.9236488
	MFE	30000	30000	30000	30163.16667	30038.8
9	Best	82.58144051	183.9047143	174.6873065	189.5462566	11.82239455
	Worst	275.100287	283.1739045	249.2618453	242.3410246	44.09695302
	Mean	211.4806106	238.8109701	203.3125985	216.0038257	29.26076862
	STD	41.58262643	25.05505101	17.48088927	12.57640716	7.210275076
	MFE	30000	30000	30000	30246.93333	30033.6
10	Best	1.340421288	0.01602575	4.51465E-09	11.80926578	9.14087E-07
	Worst	19.96317829	19.96048248	0.931304602	13.48685262	1.87018E-05
	Mean	3.544848527	6.029369132	0.062087072	12.71855001	4.61579E-06

	STD	4.523133052	8.827128996	0.236279524	0.503095444	3.42908E-06
	MFE	30000	30000	30000	30174.03333	30034.8
11	Best	3.87024E-13	0.000684082	4.67404E-14	18.67629168	5.55651E-11
	Worst	0.070984139	0.741672368	0.569327929	68.09827573	0.343918782
	Mean	0.016380538	0.480608522	0.125402819	41.81819117	0.093842602
	STD	0.016674644	0.220909193	0.12739394	11.89438954	0.089296594
	MFE	30000	30000	30000	30176.23333	30034.8
12	Best	2.95944E-12	0.101041766	0.320579961	1112.053147	0.031510408
	Worst	25.77634972	15.3687038	2.587976377	219089.505	2.00220998
	Mean	3.326291084	5.096597329	0.818413527	48693.6658	0.37438139
	STD	5.791910477	3.953017932	0.57337885	53495.37265	0.390735094
	MFE	30000	30000	30000	30234.96667	30038.6
13	Best	1.46599E-12	4.87385E-12	2.56084E-17	75765.28964	1.87724E-08
	Worst	40.25456675	48.02336319	0.09737116	4583252.205	0.240192154
	Mean	8.886192319	4.13473568	0.011985054	1560315.509	0.02981008
	STD	12.0496735	11.2745274	0.025838424	994963.699	0.062183659
	MFE	30000	30000	30000	30238.76667	30035
14	Best	0.998003838	0.998003838	0.998003839	0.998055928	0.998003838
	Worst	0.998003838	0.998004194	0.999925881	3.968250346	0.998003843
	Mean	0.998003838	0.998003852	0.998257477	1.808262855	0.998003838
	STD	1.23698E-16	6.49804E-08	0.000527552	0.810513825	9.98569E-10
	MFE	30000	30000	30000	30199.9	30035.2
15	Best	0.000307486	0.000307486	0.000324243	0.001364568	0.000424113
	Worst	0.020434946	0.008333703	0.001272374	0.009562903	0.001380486
	Mean	0.00454563	0.001289295	0.000596688	0.004038841	0.000701339
	STD	0.008058756	0.001977607	0.000244477	0.002218998	0.000213724
	MFE	30000	30000	30000	30297.76667	30030
16	Best	– ~~1.031628054 1.031628233 1.03162617 1.031552471~~ 1.031628054	– 1.031628233	– 1.03162617	– 1.031552471	– 1.031627676
	Worst	– ~~1.031584914 1.03155237 1.031600346 1.011904581~~ 1.031584914	– 1.03155237	– 1.031600346	– 1.011904581	– 1.031602254
	Mean	– ~~1.031611222 1.031611907 1.031611279 1.028544595~~ 1.031611222	– 1.031611907	– 1.031611279	– 1.028544595	– 1.031615743
	STD	1.08599E-05	1.47749E-05	7.75488E-06	0.003767043	8.22898E-06
	MFE	7041.333333	7202	8500	30176.5	2338
17	Best	0.397894345	0.397887438	0.397897956	0.397910357	0.397888025
	Worst	0.397999462	0.397996151	0.397988112	0.457108975	0.397998161
	Mean	0.397945174	0.397940169	0.39794788	0.407336514	0.397952448
	STD	2.93566E-05	3.44684E-05	2.58238E-05	0.012636874	3.35746E-05
	MFE	595.3333333	465.3333333	896	29274.23333	1524

18	Best	3	3	3.00001234	3.00144078	3
	Worst	3	3	3.002994418	3.461306359	3
	Mean	3	3	3.00043706	3.132162867	3
	STD	1.90941E-14	1.4162E-14	0.000581165	0.111886989	2.35699E-13
	MFE	2750.666667	6644	30000	30283.56667	13147.6
19	Best	–~~3.862647264-3.862646836-3.862630337-3.86273971-~~3.862647264	–3.862646836	–3.862630337	–3.86273971	–3.862476872
	Worst	–~~3.860015745-3.860014013-3.860166138-3.814862203-~~3.860015745	–3.860014013	–3.860166138	–3.814862203	–3.860018537
	Mean	–~~3.861284579-3.860875224-3.861230723-3.847444698-~~3.861284579	–3.860875224	–3.861230723	–3.847444698	–3.861305717
	STD	0.000759734	0.000609944	0.000769165	0.01227268	0.000744026
	MFE	526	326.6666667	721.3333333	29257.36667	1496
20	Best	–~~3.321514906-3.321517556-3.321340804-3.232776201-~~3.321514906	–3.321517556	–3.321340804	–3.232776201	–3.3216568
	Worst	–~~3.190272286-3.20310205-3.20310205-2.774548607-~~3.190272286	–3.20310205	–3.20310205	–2.774548607	–3.18590451
	Mean	–~~3.271481418-3.27357853-3.257887186-2.964897304-~~3.271481418	–3.27357853	–3.257887186	–2.964897304	–3.283422687
	STD	0.058118203	0.058528253	0.059569551	0.119154691	0.056560408
	MFE	15438	12787.33333	16832	30262.93333	14456.4
21	Best	–~~10.15319968-10.15319968-10.15319968-9.237961427-~~10.15319968	–10.15319968	–10.15319968	–9.237961427	–10.15319968
	Worst	–~~4.051730311-2.630471668-2.630471668-2.348276139-~~4.051730311	–2.630471668	–2.630471668	–2.348276139	–3.873011974
	Mean	–~~7.571532266-7.286516369-7.988655139-5.13102673-~~7.571532266	–7.286516369	–7.988655139	–5.13102673	–8.758677601
	STD	2.183528248	2.791706593	2.300114085	2.319287801	2.257015113
	MFE	30000	30000	30000	30201.03333	30029.6
22	Best	–~~10.40293072-10.40293811-10.40293612-10.26936583-~~10.40293072	–10.40293811	–10.40293612	–10.26936583	–10.40292495
	Worst	–~~3.724300347-1.837592971-7.655316059-2.356385661-~~3.724300347	–1.837592971	–7.655316059	–2.356385661	–4.785539658
	Mean	–~~8.513729479-9.193001948-10.14971362-5.773917362-~~8.513729479	–9.193001948	–10.14971362	–5.773917362	–9.962990885
	STD	2.516505404	2.672409474	0.667381659	2.772503044	1.251621589
	MFE	19810	7462.666667	10530.66667	30195.36667	28090
23	Best	–~~10.53640962-10.53640895-10.53640895-9.998537379-~~10.53640962	–10.53640895	–10.53640895	–9.998537379	–10.53640573
	Worst	–~~5.032711076-2.421734027-2.4273352-2.420451607-~~5.032711076	–2.421734027	–2.4273352	–2.420451607	–3.835426802
	Mean	–~~9.76293601-8.189952935-9.457598582-5.207676489-~~9.76293601	–8.189952935	–9.457598582	–5.207676489	–10.10837656
	STD	1.635492849	3.653678401	2.385255584	2.37062617	1.331195279

	MFE	18558.66667	10478	11730.66667	30157.76667	27428.6
	FMR	2.43	3.17	2.13	4.83	1.52
	Rank	3	4	2	5	1

Based on the two metrics, mean solution and STD, for the seven unimodal functions, ID = 1 to 7, the proposed ERA commonly outperforms all the other algorithms for the four functions with ID = 4, 5, 6, and 7. Unfortunately, it is worse than Rao-3 and Rao-1 for two functions with ID = 1 and 2. Besides, it is much worse than Rao-1 for the function ID = 3.

Next, the investigation on the six multimodal functions, ID = 8 to 13, informs that the proposed ERA also generally outperforms the competitors, where it achieves much lower mean solutions for three functions with ID = 9, 10, and 12. It is slightly worse than Rao-3 and Rao-2 for the function ID = 8. It is much worse than Rao-1 and Rao-3 for the function ID = 11 and 13, respectively.

Finally, the investigation on the ten low-dimension multimodal functions, ID = 14 to 23, shows that the proposed ERA mostly gives better or equal mean solutions than the competitors. It reaches the best solutions for the three benchmark functions with ID = 20, 21 and 23. It gives the same or similar global solutions, with quite low MFE, as the three Rao algorithms for the benchmark function with ID = 16, 17, 18, and 19. It is slightly worse than Rao-1 or Rao-3 only for three benchmark functions (ID = 14, 15, and 22).

As a summary, based on Table 3, ERA reaches better mean solutions than all the competitors for 10 benchmark functions. It gives the same and worse mean solutions for 4 and 9 benchmark functions, respectively. Statistically, based on the Friedman mean rank (FMR), ERA gives the highest performance with the lowest FMR of 1.52. The Wilcoxon rank-sum test (WRST) illustrated in Table 4 confirms that ERA is significantly better than all the competitors for the six benchmark functions (ID = 4, 7, 9, 10, 12, and 23), where all the p-values are less than 0.05. Meanwhile, for the four benchmark functions (ID = 5, 6, 20, and 21), ERA is only significantly better than some competitors but not for the others.

Table 4

The table layout displayed in this section is not how it will appear in the final version. The representation below is solely purposed for providing corrections to the table. To preview the actual presentation of the table, please view the Proof.

The p-values of Wilcoxon rank sum test (WRST) for 23 classic benchmark functions.

ID	ERA vs Rao-1	ERA vs Rao-2	ERA vs Rao-3	ERA vs FA-CL
1	3.68973E-11	3.01986E-11	3.01986E-11	3.01986E-11
2	8.89099E-10	3.01986E-11	3.01986E-11	3.01986E-11
3	3.01986E-11	3.01986E-11	3.01986E-11	3.01986E-11
4	3.01986E-11	3.01986E-11	4.57257E-09	3.01986E-11
5	0.027086318	0.115362360	0.000526404	3.01986E-11
6	5.29270E-12	0.002309997	0.537496020	5.18120E-12
7	3.33839E-11	3.33839E-11	0.022360148	3.01986E-11
8	0.000421751	0.105469947	0.065671258	3.01986E-11
9	3.01986E-11	3.01986E-11	3.01986E-11	3.01986E-11
10	3.01986E-11	3.01986E-11	8.48477E-09	3.01986E-11
11	8.66343E-05	2.01522E-08	0.501143668	3.01986E-11
12	0.040595001	1.28704E-09	3.83067E-05	3.01986E-11

13	0.009883401	0.013271805	0.001766564	3.01986E-11
14	0.405861585	9.89193E-09	6.4749E-120	5.21903E-12
15	0.318136088	0.529748183	0.074827008	3.33839E-11
16	0.093340797	0.420386330	0.055545693	3.01986E-11
17	0.437641335	0.157975689	0.510597937	4.19968E-10
18	0.036392066	0.812931300	3.01041E-11	3.01041E-11
19	0.935191970	0.022360148	0.641423523	1.42942E-08
20	0.369977675	0.110560585	0.659705270	7.38908E-11
21	0.283376373	0.425345373	0.578792661	4.44405E-07
22	0.923442132	0.000556012	5.97056E-05	1.28704E-09
23	0.003670893	0.019094054	0.001235991	4.19968E-10

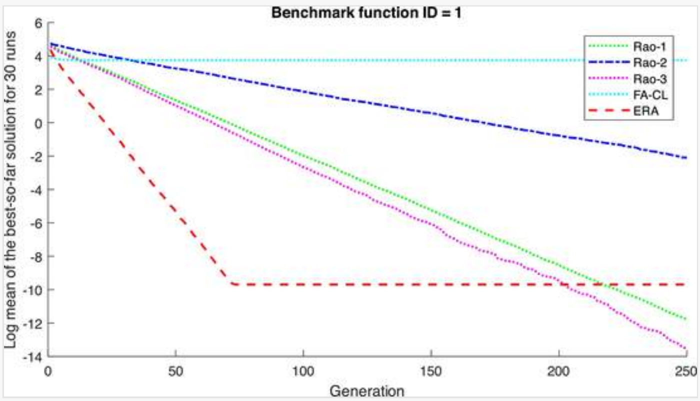
Moreover, the detailed investigations are then provided by the convergence curve analysis. The three subsections below discuss the convergence curves in detail for three benchmark groups: high-dimensional unimodal, high-dimensional multimodal, and low-dimensional multimodal.

3.3.1 Investigation on 30-dimensions unimodal functions

A detailed investigation of the seven 30-dimensions unimodal benchmark functions, ID = 1 to 7, is discussed by illustrating two convergence analyses of the proposed ERA and all the competitors. For each benchmark function, the maximum number of function evaluations is set to 30,000 with 30 runs to reduce the coincidence.

Fig. 6 shows the evolution of all the algorithms until convergence to the optimum solution for the benchmark function of Sphere (ID = 1). The horizontal axis is the generation, calculated as 30,000 function evaluations divided by the population size p. The random seeds of the 30 initial populations are the same for the algorithms that use the same optimum population size p. Hence, in this case, Rao-1, Rao-2, and FA-CL use the same initial population since they have the same optimum $p = 20$. In contrast, the Rao-3 and ERA use a different initial population because they have the optimum $p = 40$ and $p = 60$, respectively. Due to the different optimum p for each algorithm, the evolution is illustrated using the different step sizes of generation to get fairness. Here, the proposed ERA uses a step size of 2, Rao-3 uses 3, and the rests use 6 so that all the algorithms show the same generations of 1 to 250. It can be seen in Fig. 6 that the ERA is worse than Rao-1 and Rao-3. This result also applies to two other similar unimodal functions ID = 2 and 3.

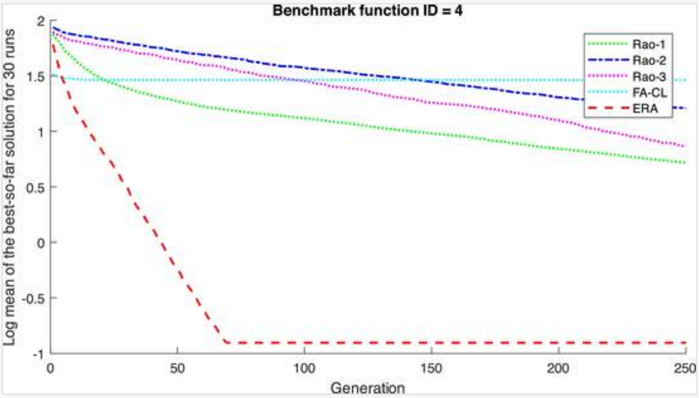
Fig. 6



Convergence analysis for a unimodal benchmark function of Sphere (ID = 1).

Fig. 7 shows the evolution of all the algorithms for the benchmark function of Schwefel 2.21 (ID = 4). ERA converges much faster than the others. It converges in the one-fourth of the evolution, and, at the end of evolution, it gives the lowest mean solution compared to Rao-1, Rao-2, Rao-3, and FA-CL that produce much worse solutions. Similar results also happen to three other unimodal functions ID = 5, 6, and 7.

Fig. 7

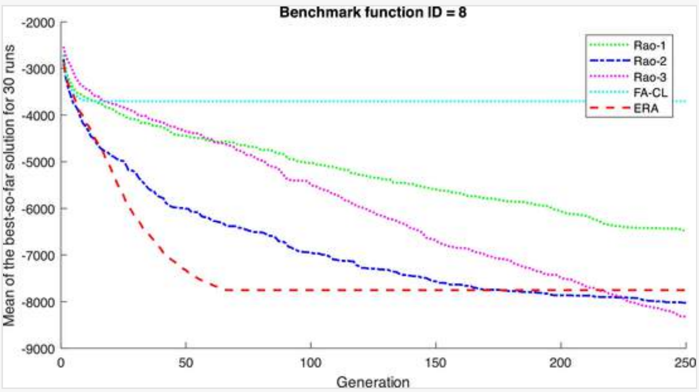


Convergence analysis for a unimodal benchmark function of Schwefel 2.21 (ID = 4).

3.3.2 Investigation on 30-dimensions multimodal functions

Next, a detailed investigation of the six 30-dimensional multimodal benchmark functions, ID = 8 to 13, is illustrated by two convergence analyses of the proposed ERA and all other algorithms. Fig. 8 shows the convergence curves for the multimodal function of Schwefel (ID = 8) that has many local minima. ERA performs a little worse than Rao-1 and Rao-3, where it converges to a slightly bigger solution. This result also applies to two other multimodal functions (ID = 11 and 13).

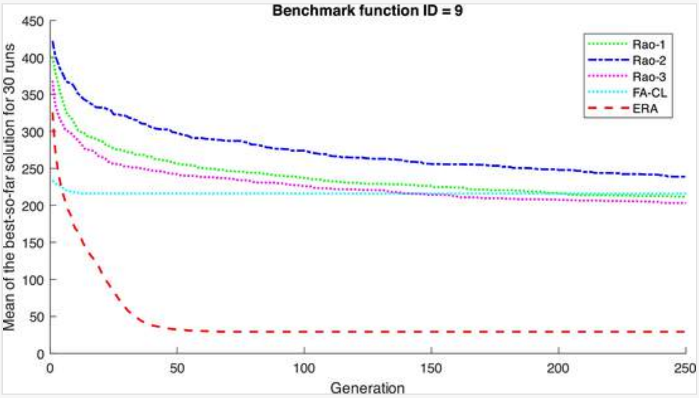
Fig. 8



Convergence analysis for a multimodal benchmark function of Schwefel (ID = 8).

Next, the convergence analysis is provided for the multimodal function of Rastrigin with ID = 9 that also has many local minima. Fig. 9 illustrates that the ERA converges much faster than the others. It evolves quickly in the beginning generations and gives the lowest mean solution among the competitors at the end of evolution. ERA also converges similarly for two other unimodal functions with ID = 10 and 12.

Fig. 9

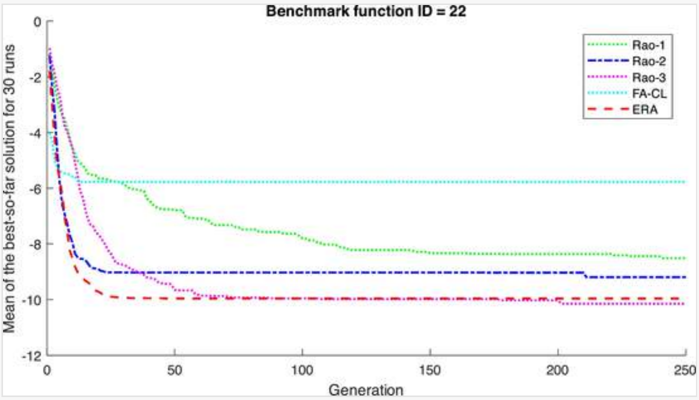


Convergence analysis for a multimodal benchmark function of Rastrigin (ID = 9).

3.3.3 Investigation on low-dimension multimodal functions

Finally, the detailed investigations of ten benchmark low-dimension multimodal functions, ID = 14 to 23, are also illustrated by some convergence analyses of ERA and the competitors. Fig. 10 shows the evolution of all the algorithms for the 4-dimensions multimodal function of Shekel 7 (ID = 22) that has broad flat areas. In this case, ERA converges to a similar solution to Rao-3.

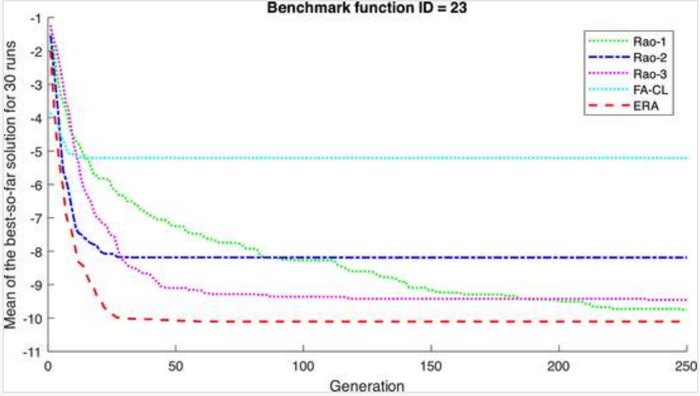
Fig. 10



Convergence analysis for a multimodal benchmark function of Shekel 7 (ID = 22).

Furthermore, the convergence analysis is carried out for the 4-dimensions multimodal function of Shekel 10 (with ID = 23) with broad flat areas. Fig. 11 shows that ERA performs the best evolution and converges to a better solution than the competitors. This result also applies to three other low-dimensions multimodal functions with ID = 14, 15, and 21. Meanwhile, ERA gives the same (or similar) convergence curves as the competitors for 16, 17, 18, 19, and 20.

Fig. 11



Convergence analysis for a multimodal benchmark function of Shekel 10 (ID = 23).

Those results of FMR, WRST, and convergence curves indicate that ERA generally outperforms all the competitors. It proves that the proposed schemes: two sub-populations and evolutionary operators equipped with the adaptation procedure can effectively control the exploration and exploitation balance. The detailed investigations on the fitness-based adaptation scheme will be given in Section 3.6.

3.4 Evaluation on CEC-C06 2019

The CEC-C06 2019 is a set of ten modern benchmark functions, namely CEC01, CEC02,..., CEC10. As described in [30], all the functions are scalable. The seven functions (CEC04 to CEC10) are shifted and rotated, but the others (CEC01 to CEC03) are not. Those seven functions are set as 10-dimensional minimization problems in the interval [−-100, 100] while the rests have different dimensions of 9, 16, and 18 in the interval [−-8192, 8192], [-16384, 16384], and [-8192, 8192], [−16384, 16384], and [−4, 4], respectively. Besides, all ten benchmarks have the same global optimum of 1.

The proposed ERA is evaluated using those ten CEC-C06 2019 benchmarks, where their Matlab codes refer to [30], to see its ability to handle the effects of shift and rotation of the test functions. It is also compared with the four other algorithms in their best performances using the parameter settings described in Section 3.2. All algorithms are run 30 times with 30,000 function evaluations each to get meaningful statistical results. Moreover, both FMR and WRST with the p-values are also provided. The experimental results illustrated in Table 5 show that ERA outperforms all the competitors for most benchmark functions. It reaches significantly better mean solutions for 7 out of 10 benchmarks: CEC01, CEC03, CEC05, CEC06, CEC07, CEC08, and CEC10. It gives little worse solutions than Rao-1 and Rao-2 for only three benchmarks: CEC02, CEC04, and CEC09. The Friedman mean rank shows that ERA is the first rank, where it achieves the lowest FMR of 1.50. Meanwhile Rao-1, Rao-2, Rao-3, and FA-CL give much worse FMR of 2.40, 2.50, 3.50, and 5.00, respectively.

Table 5

The table layout displayed in this section is not how it will appear in the final version. The representation below is solely purposed for providing corrections to the table. To preview the actual presentation of the table, please view the Proof.

Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA for ten benchmarks of CEC-C06 2019IDMetricRao-1Rao-2Rao-3FA-CLERA.

ID	Metric	Rao-1	Rao-2	Rao-3	FA-CL	ERA
CEC01	Best	390354788.1	2916410938	1456657938	20718576008	25205057.57
	Worst	18579735758	26602967009	22606057356	3.52496E+11	4087331440
	Mean	2969561815	12158813001	8150847618	1.24685E+11	873534390.1
	STD	3458651870	6343181227	6050933807	88017205415	854049190.6

	MFE	30000	30000	30000	30258.9	30034.8
CEC02	Best	17.34285714	17.34285714	17.39624973	985.6431507	17.34385166
	Worst	17.34285714	17.34285714	17.46552571	3599.0291	17.38067398
	Mean	17.34285714	17.34285714	17.43100117	2309.760136	17.35587136
	STD	7.04391E-15	6.66287E-15	0.019863442	686.3489262	0.008533771
	MFE	30000	30000	30000	30167.6	30050
CEC03	Best	12.70240422	12.70240422	12.70240422	12.70243367	12.70240422
	Worst	12.70251646	12.70252446	12.70253809	12.70313897	12.70240457
	Mean	12.70241519	12.70242315	12.70243599	12.70275408	12.70240423
	STD	2.61665E-05	3.04861E-05	3.51307E-05	0.000177661	6.36292E-08
	MFE	30000	30000	30000	30194.93333	30036.4
CEC04	Best	28.09976484	30.65215071	161.5987987	1416.884608	12.6848588
	Worst	55.30958498	67.28669199	293.2794152	7138.324036	137.9261529
	Mean	39.24039894	48.18340414	214.1502774	3937.361268	42.30643978
	STD	6.336978985	8.897804689	35.66651224	1184.043609	30.04084127
	MFE	30000	30000	30000	30215.03333	30032.4
CEC05	Best	1.280029538	1.40996188	1.5745926	1.943716683	1.025782758
	Worst	1.698018753	2.006177027	1.97844018	2.898249478	1.401388266
	Mean	1.521217618	1.676196554	1.810647002	2.607524	1.152216958
	STD	0.129249904	0.115365294	0.087314766	0.244163606	0.089725781
	MFE	30000	30000	30000	30205.93333	30042
CEC06	Best	9.733872974	9.387891192	8.762132719	10.44553117	9.112980123
	Worst	11.45221129	11.28957571	11.75705625	13.24136925	10.88436736
	Mean	10.56696371	10.46733143	10.51740995	12.09261107	10.03315348
	STD	0.440170202	0.509698432	0.686756457	0.804138499	0.496448922
	MFE	30000	30000	30000	30177.5	30040.4
CEC07	Best	272.7315304	155.7756802	286.8881473	349.6263842	155.6301339
	Worst	909.7970404	873.1138007	971.9938573	1424.750579	623.6837399
	Mean	621.4329426	523.9535639	619.1594318	982.5206281	397.7660801
	STD	182.4620119	177.2735789	175.8854884	241.0657041	115.7688378
	MFE	30000	30000	30000	30189.7	30055.6
CEC08	Best	5.3786702	5.158716685	4.494612984	5.437849384	2.610104398
	Worst	6.875548609	6.47014268	6.121359357	7.20917994	5.372692569
	Mean	6.000536425	5.812160004	5.518584178	6.554942645	4.300037284
	STD	0.355903577	0.33238346	0.394227163	0.415653324	0.742690621
	MFE	30000	30000	30000	30155.36667	30048.2
CEC09	Best	2.344511638	2.410388957	8.415440069	107.8777016	2.471916423
	Worst	2.364279095	2.678458436	106.5608922	876.6191415	18.25738167
	Mean	2.352775129	2.519079794	47.78092232	560.843819	4.325248722

	STD	0.004691204	0.063901687	21.70671343	192.3434718	2.804045367
	MFE	30000	30000	30000	30187.13333	30044.2
CEC10	Best	20.14320415	20.24340503	20.12140073	20.09700221	20.14872943
	Worst	20.57898412	20.52293164	20.59178469	20.77386591	20.49628839
	Mean	20.42696562	20.40786804	20.42908964	20.60276541	20.34605205
	STD	0.082810151	0.073689655	0.085669202	0.131641593	0.072081928
	MFE	30000	30000	30000	30155.63333	30027.6
	FMR	2.40	2.50	3.50	5.00	1.50
	Rank	2	3	4	5	1

Moreover, the Wilcoxon rank-sum test illustrated in Table 6 confirms that ERA is significantly better than all the competitors for the seven benchmark functions, where all the p-values are lower than the significance level of 0.05, except for the CEC03 where ERA is not significantly better than Rao-1. In contrast, for CEC02, ERA is slightly worse than Rao-1 with a p-value of bigger than 0.05. Meanwhile, for CEC04 and CEC09, ERA is much worse than Rao-1 and Rao-2 with p-values of less than 0.05.

Table 6

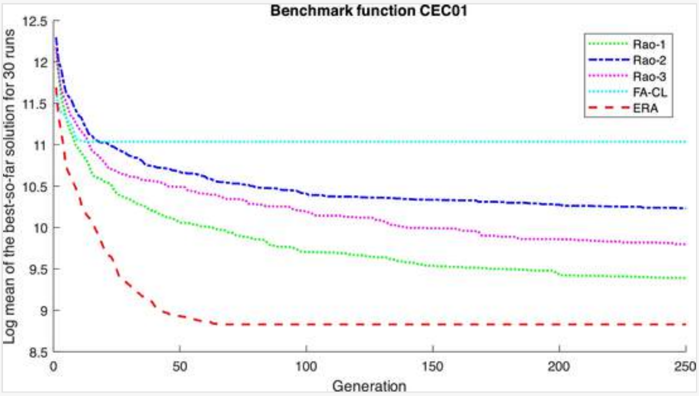
The table layout displayed in this section is not how it will appear in the final version. The representation below is solely purposed for providing corrections to the table. To preview the actual presentation of the table, please view the Proof.

The p-values of Wilcoxon rank sum test (WRST) for ten benchmark functions of CEC-C06 2019.

ID	ERA vs Rao-1	ERA vs Rao-2	ERA vs Rao-3	ERA vs FA-CL
CEC01	1.33668E-05	3.68973E-11	6.72195E-10	3.01986E-11
CEC02	2.36384E-12	6.31878E-12	3.01986E-11	3.01986E-11
CEC03	0.065461305	1.01761E-05	1.77301E-09	3.01986E-11
CEC04	0.053685253	0.004856016	3.01986E-11	3.01986E-11
CEC05	8.99341E-11	3.01986E-11	3.01986E-11	3.01986E-11
CEC06	0.000212646	0.002156638	0.001370333	9.91863E-11
CEC07	1.09069E-05	0.005322078	4.7445E-06	2.37147E-10
CEC08	3.01986E-11	7.38908E-11	4.99795E-09	3.01986E-11
CEC09	3.01986E-11	4.19968E-10	3.68973E-11	3.01986E-11
CEC10	5.60728E-05	0.00185748	3.59234E-05	7.38029E-10

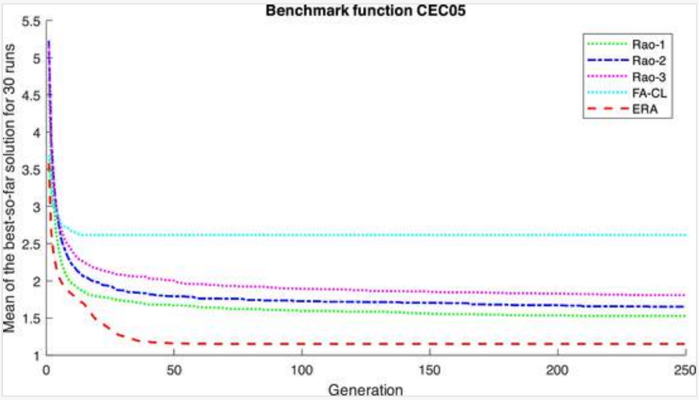
Furthermore, the detailed investigations are then discussed by illustrating the convergence analysis of ERA and all the competitors. For each benchmark function, the maximum number of function evaluations is set to 30,000 with 30 runs. Fig. 12 illustrates the evolutionary processes of all algorithms until converge the optimum solution for CEC01. In this case, ERA converges to a much better solution than the other algorithms. ERA also gives similar curves for six other benchmarks: CEC03, CEC05, CEC06, CEC07, CEC08, and CEC10. Figs. 13 and 14 illustrate the converge curves for CEC05 and CEC08. Impressively, for CEC05, ERA evolves most quickly in the beginning generations and finally gives the best mean solution of 1.152216958, which is quite close to the known global optimum of 1.

Fig. 12



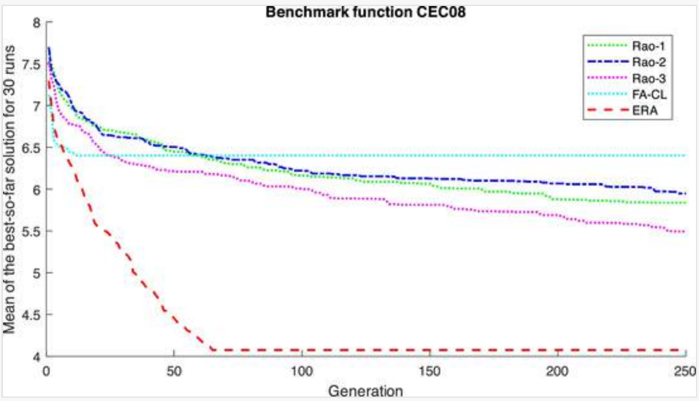
Convergence analysis for CEC01.

Fig. 13



Convergence analysis for CEC05.

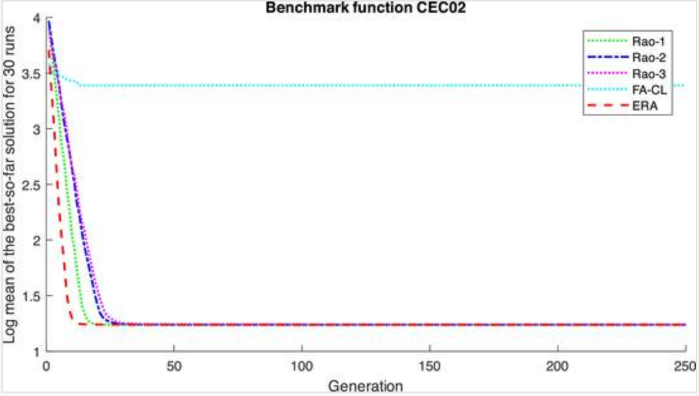
Fig. 14



Convergence analysis for CEC08.

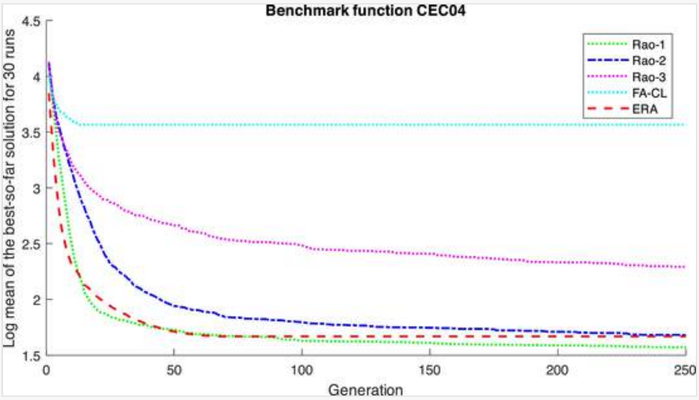
Next, the detailed investigation is then carried out for CEC02. Fig. 15 shows that ERA converges to a similar solution to Rao-1, Rao-2, and Rao-3. In addition, ERA also gives similar curves for two other benchmarks: CEC04 and CEC09, as illustrated in Figs. 16 and 17.

Fig. 15



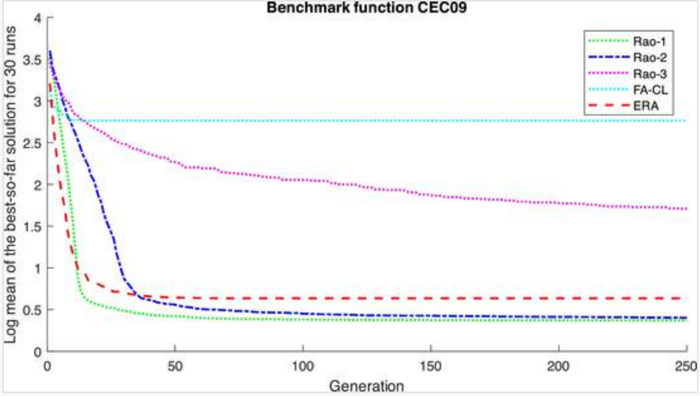
Convergence analysis for CEC02.

Fig. 16



Convergence analysis for CEC04.

Fig. 17



Convergence analysis for CEC09.

The results of FMR, WRST, and convergence curves above indicate that ERA is generally better than all the competitors in handling the effects of shift and rotation. It can be achieved since ERA is designed with two schemes: two sub-populations and evolutionary operators. Besides, it is also equipped with a fitness-based adaptation scheme to dynamically tune the three sensitive parameters: s, a, and b throughout the evolution process, which effectively controls the exploration and exploitation balance in searching for the global optimum solution. A detailed investigation will be provided in Section 3.6.

3.5 Evaluation on real world problems

Finally, ERA is evaluated using the global trajectory optimization problems (GTOP), which European Space Agency provides. Here, five real-world problems: Cassini1, GTOC1, Messenger, Sagas, and Cassini2 are used as the benchmarks to examine its capability to tackle the constrained problems. Here, the four cases: Cassini1, Messenger, Sagas, and Cassini2 are the minimization problems while GTOC1 is a maximization. Here, the summary of those five problems is given briefly; the more detailed descriptions and their Matlab codes can be seen in [31,33,34].

As described in [31], Cassini1 is a mission of multiple gravity assist (MGA) without the possibility of using deep space manouvres, which is related to the Cassini spacecraft trajectory design problem. The objective is to minimize the total delta velocity ΔV accumulated during the mission with some given constraints, where the planetary fly-by sequence is Earth-Venus-Venus-Earth-Jupiter-Saturn. It has six dimensions with a known global minimum solution of 4.93 km/see.

GTOC1 is also an MGA problem, where the objective is to maximize the change in the semi-major axis of the asteroid orbit following an anaelastic impact of the spacecraft with the asteroid under several given constraints. It has 8 dimensions and the known global maximum is $f_{\max} = 1,580,599$ kg km^2/s^2. In this paper, GTOC1 is converted into a minimization problem by modifying the objective function to be $f_{\min} = 1,580,599 - f_{\max} = 0$ to make it the same as all the other problems and simpler in the further evaluation.

Furthermore, Messenger, Sagas, and Cassini2 are the MGA problems with the possibility of using deep space manouvres (MGA-1DSM). The objective of Messenger is to minimize the total delta velocity ΔV. It has 18 dimensions, and the global minimum solution is 8.70257 km/see. Meanwhile, the objective of Sagas is to minimize the overall mission length of fly-by Jupiter and reach 50AU. It is a 12-dimensional problem with a global minimum of 18.1923 years. Finally, Cassini2 is similar to Cassini1, but it is a bigger 22-dimensional problem with much higher complexity. The known best solution is 8.92401 km/see.

In the five real-world problems, ERA is also compared with the four other algorithms in their best performances using the parameter settings described in Section 3.2. All algorithms are run 30 times with 200,000 function evaluations each as used in [31]. In addition, both FMR and WRST with the p-values are also provided to confirm the significance of their performance. The experimental results illustrated in Table 7 show that ERA outperforms the competitors for three out of five problems. Hence, the Friedman mean rank places ERA in the first rank with the lowest FMR of 1.80. Unfortunately, the Wilcoxon rank-sum test illustrated in Table 8 indicates that ERA significantly outperforms (with p-values of less than 0.05) some of the competitors. It is slightly (not significantly) better than Rao-1, Rao-2, and Rao-3 for Cassini1, GTOC1, and Sagas, respectively.

Table 7

The table layout displayed in this section is not how it will appear in the final version. The representation below is solely purposed for providing corrections to the table. To preview the actual presentation of the table, please view the Proof.

Comparison of Rao-1, Rao-2, Rao-3, FA-CL, and ERA [Instruction: "for global" --> "for five global"]for global trajectory optimization problems.

Problem	~~MetricRao-1Rao-2Rao-3FA-CLERA~~Metric	Rao-1	Rao-2	Rao-3	FA-CL	ERA
Cassini1	Best	5.303768392	4.936855136	5.61179549	7.074851303	5.612797557
	Worst	20.07805387	25.93297419	15.77360635	26.20831412	14.83107214
	Mean	11.87916323	12.12477947	12.56694735	16.6899857	10.61999692
	STD	3.554957586	3.64432772	2.23350672	4.840145141	2.876504936
	MFE	200000	200000	200000	200187.4333	200044.6
GTOC1	Best	631827.7829	458914.6802	376896.2971	1411461.987	710462.7745
	Worst	1548886.166	1551541.426	1536925.137	1569631.301	1311152.824
	Mean	1147746.121	1093465.404	1051211.872	1524073.714	956084.4087
	STD	233788.8767	331123.7832	266223.3551	43045.94967	136868.8532
	MFE	200000	200000	200000	200124.6333	200049.6
Messenger	Best	12.41560806	11.27514945	11.99302003	20.78291121	14.33306793
	Worst	25.50492428	20.425443	22.54997489	28.24295896	20.46142743
	Mean	18.35040915	14.97590782	16.61688953	25.51273265	16.37566883
	STD	3.328431397	2.476903152	2.802312585	2.076873921	1.412144879
	MFE	200000	200000	200000	200136.7	200047.8
Sagas	Best	76.02938156	322.8420001	246.9977338	1267.982722	504.8182077
	Worst	1615.462402	3734.488398	2310.454904	2319.048595	973.1650916
	Mean	1024.564265	1202.784813	1017.331876	1722.402349	927.9858521
	STD	345.5888343	619.2934569	310.6547919	205.0753697	99.79849535
	MFE	200000	200000	200000	200206.9	200045.6
Cassini2	Best	10.62971719	11.03729327	15.93387459	32.97381024	21.03860384
	Worst	44.68073676	36.96644582	32.11412822	47.76452256	30.01962572
	Mean	25.0919048	23.93646566	25.58562866	40.28520967	26.03675587
	STD	8.783109087	6.801655034	3.476676083	4.082917564	2.049424528
	MFE	200000	200000	200000	200088.5667	200036.6

	FMR	3.00	2.40	2.80	5.00	1.80
	Rank	4	2	3	5	1

Table 8

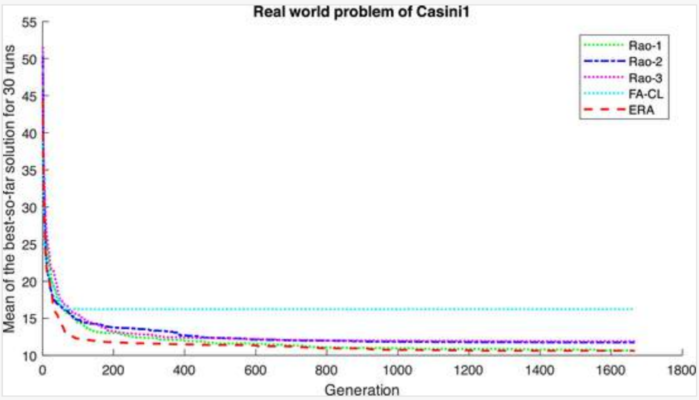
The table layout displayed in this section is not how it will appear in the final version. The representation below is solely purposed for providing corrections to the table. To preview the actual presentation of the table, please view the Proof.

The p-values of Wilcoxon rank sum test (WRST) for five global trajectory optimization problems.

Problem	ERA vs Rao-1	ERA vs Rao-2	ERA vs Rao-3	ERA vs FA-CL
Cassini1	0.318304227	0.072445596	0.00033679	8.1975E-07
GTOC1	0.000356384	0.108689773	0.02812867	3.01986E-11
Messenger	0.005084222	0.000691252	0.437641335	3.01986E-11
Sagas	0.022360148	0.000168132	0.111986872	3.01986E-11
Cassini2	0.332854692	0.008314609	0.529782491	3.01986E-11

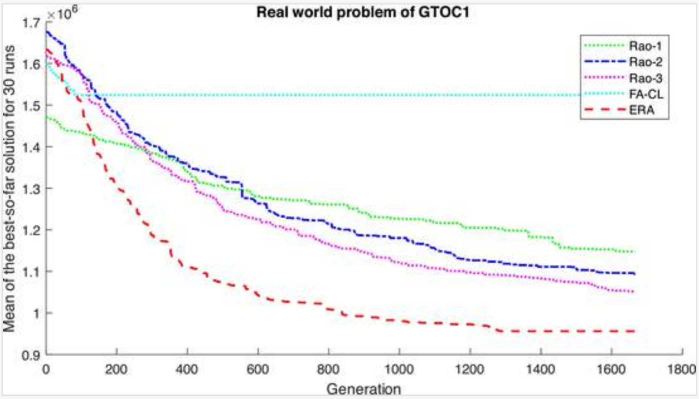
Furthermore, the detailed investigations are then discussed by illustrating the convergence curves of ERA and the competitors. Fig. 18 illustrates the evolution of all algorithms until to the optimum solution for [Instruction: Please update "CEC01" --> "Cassini1"]CEC01. In this case, ERA converges to a better solution than the other algorithms. Like all the competitors, ERA evolves quickly in the beginning generations and gets stagnation. In addition, ERA also gives similar curves for GTOC1 and Sagas, as illustrated in Figs. 19 and 21. However, for GTOC1, ERA reaches a much better solution.

Fig. 18



Convergence analysis for Cassini1.

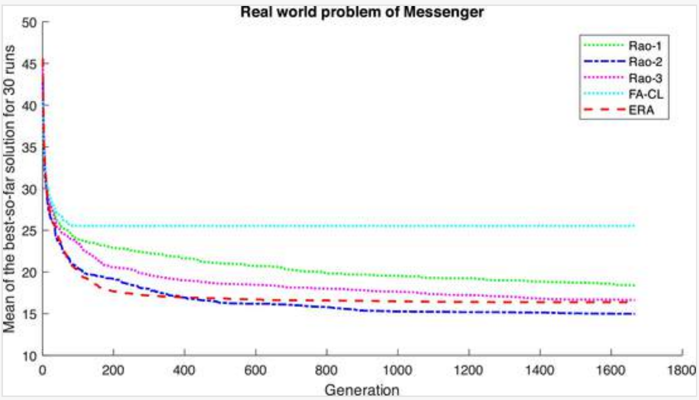
Fig. 19



Convergence analysis for GTOC1.

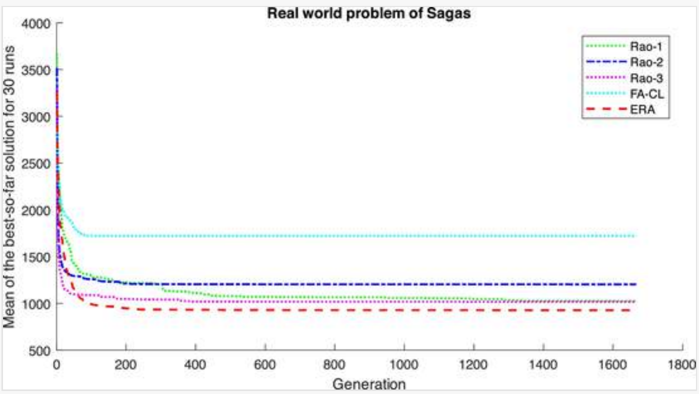
For both Messenger and Cassini2 problems, ERA gives worse convergence curves than the Rao algorithms. Fig. 20 informs that, in the beginning, ERA gives the same converge curve as the Rao-2, but it gets stuck after half of the generations while Rao-2 keeps evolving and reaches a little better solution. Meanwhile, Fig. 22 shows that ERA evolves quickly in the early generations but finally converges to a slightly worse solution than the three Rao algorithms.

Fig. 20



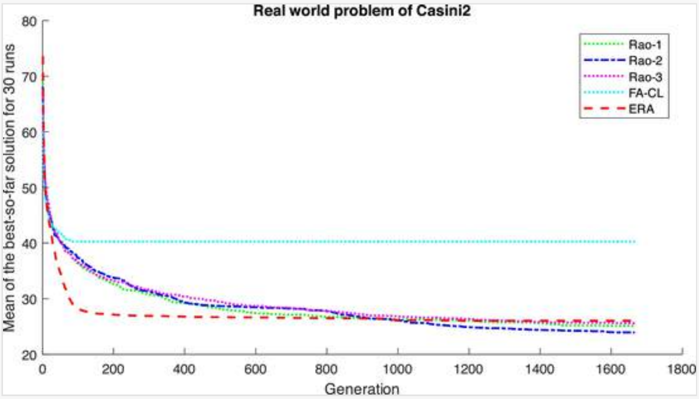
Convergence analysis for ~~Messenger~~[Instruction: Please update "messenger" --> "Messenger" since it is a named-entity for one of the real-world global trajectory optimization problems.]messenger.

Fig. 21



Convergence analysis for Sagas.

Fig. 22



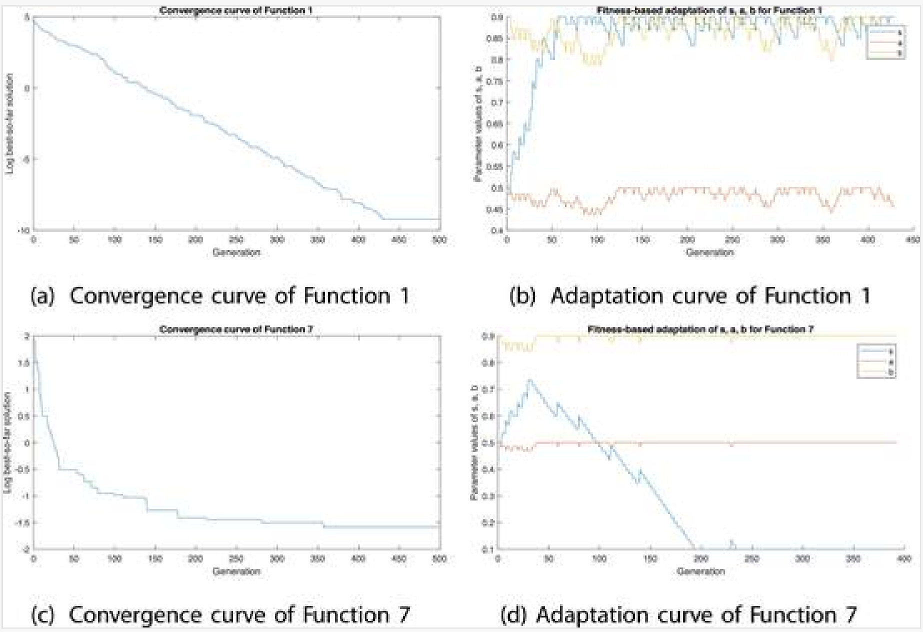
Convergence analysis for Cassini2.

However, those results of FMR, WRST, and convergence curves inform that ERA is better than the competitors in tackling the constrained real-world problems. It can be implied that the two proposed schemes: two sub-populations and evolutionary operators, as well as the introduced adaptation procedure, effectively balance the exploration and exploitation strategies. A detailed investigation regarding the adaptation scheme will be provided in Section 3.6.

3.6 Investigation on fitness-based adaptation scheme

The proposed fitness-based adaptation scheme is evaluated here using some benchmarks to see its ability to control the exploration-exploitation balance. First, two classic benchmarks (with ID = 1 and 7) are selected as the representative 30-dimensional unimodal functions. Fig. 23a illustrates that, for the Sphere function (ID = 1) that has only one global optimum, ERA converges quite fast. It can be achieved since ERA works in an exploration manner in the beginning generations and then quickly changes into an exploitation fashion, where the three parameters s, a, and b reach around the maximum values of 0.9, 0.9, and 0.5, respectively, as shown in Fig. 23b. Meanwhile, Fig. 23c illustrates that, for the Quartic function (with ID = 7) having many noises, ERA converges more slowly and gets a stagnation. In this case, ERA works in an exploration strategy throughout the evolution, where s tends to go to the minimum value of 0.1 but a and b reach the maximum values of 0.9 and 0.5, respectively, as shown in Fig. 23d.

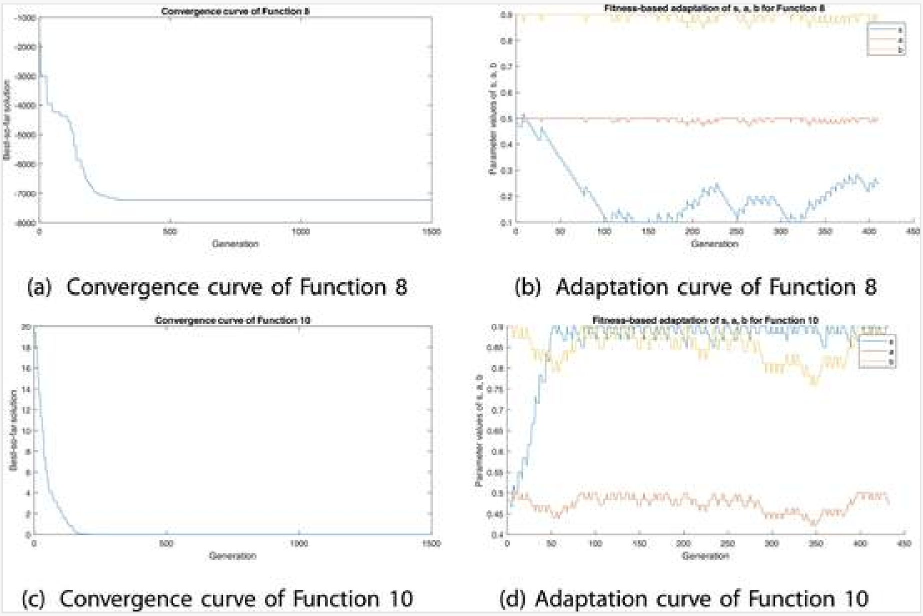
Fig. 23



Convergence and adaptation curves for 30-dimensional unimodal benchmark functions with ID = 1 and 7.

Next, two classic benchmarks with ID = 8 and 10 are selected as the representative 30-dimensional multimodal functions. Fig. 24a illustrates that, for the Schwefel function (ID = 8) having many global optimum solutions, ERA evolves quickly in early generations but finally gets stuck on a local optimum. Here, ERA works in an exploration strategy throughout the evolution, where s tends to go around the minimum value but a and b on the maximum values, as shown in Fig. 24b. Meanwhile, Fig. 24c illustrates that, for the Ackley function (ID = 10) that also has many global optimum solutions, ERA converges quickly and also gets a stagnation. In this case, ERA tends to work in an exploitation strategy throughout the evolution, where s is on the maximum value, but a and b tends on the medium values, as shown in Fig. 24d.

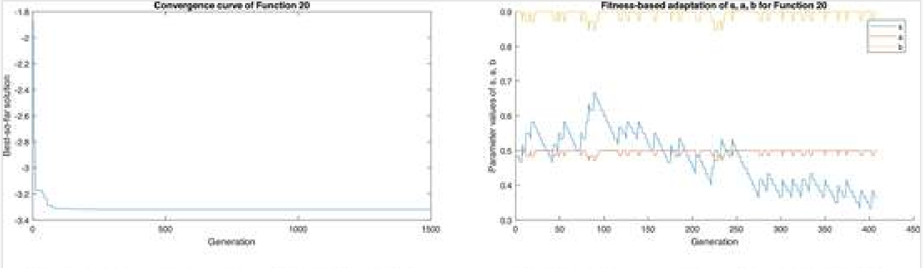
Fig. 24



Convergence and adaptation curves for 30-dimensional multimodal benchmark functions with ID = 8 and 10.

Two classic benchmarks with ID = 20 and 21 are then chosen as the representative low-dimensional multimodal functions. Fig. 25a illustrates that, for the function of Hartman 6 (ID = 20) with many global optima, ERA converges quite fast to the global optimum. It can be seen that ERA works in an exploration-exploitation balance strategy throughout the evolution, where s is fluctuating between the minimum and the medium values while a and b are around the maximum values, as shown in Fig. 25b. Meanwhile, Fig. 25c illustrates that, for the function of Shekel 5 (ID = 21), ERA converges quickly to the global optimum. It tends to work in an exploration strategy throughout the evolution since Shekel 5 has a broad flat area, where s is around the minimum value, but a and b tends on the maximum values, as shown in Fig. 25d.

Fig. 25



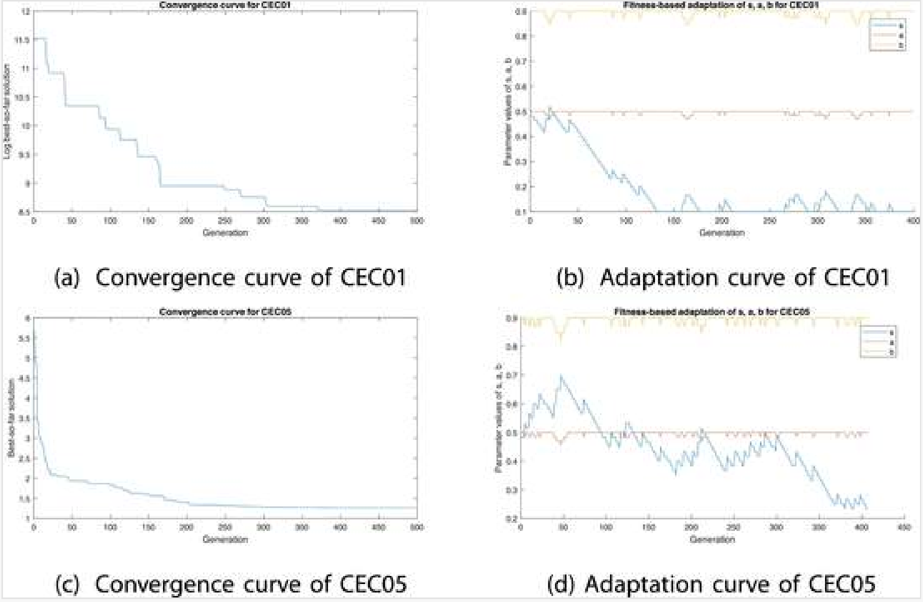
(a) Convergence curve of Function 20 (b) Adaptation curve of Function 20

(c) Convergence curve of Function 21 (d) Adaptation curve of Function 21

Convergence and adaptation curves for low-dimensional multimodal benchmark functions with ID = 20 and 21.

Two benchmarks of CEC01 and CEC05 are then chosen as the representative functions without and with both shifting and rotation, respectively. Fig. 26a illustrates that, for the function without shifting and rotation, ERA evolves slowly. Unfortunately, it gets stuck for some generations and converges to a solution far from the global optimum. It can be seen that ERA works in an exploration-exploitation balance strategy throughout the evolution, where s is fluctuating between the minimum and the medium values while a and b are around the maximum values, as shown in Fig. 26b. Meanwhile, Fig. 26c illustrates that, for the shifted and rotated function, ERA converges quickly to a solution near the global optimum of 1. It works dynamically in exploration and exploitation strategy during the evolution, where s is around the medium value, but a and b tends on the maximum values, as shown in Fig. 26d.

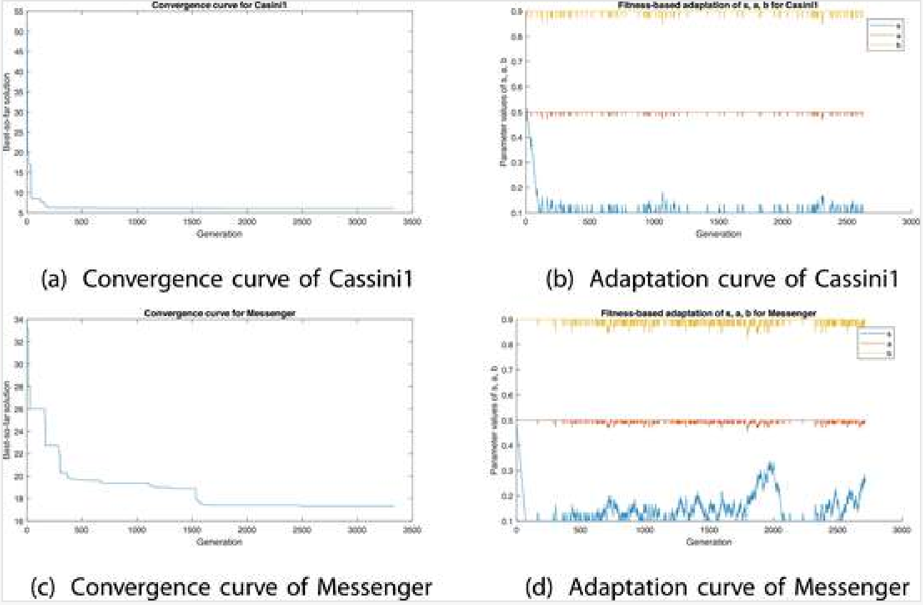
Fig. 26



Convergence and adaptation curves for CEC01 and CEC05.

Finally, both Cassini1 and Messenger are selected as the representative of the real-world problems. Fig. 27a illustrates that ERA evolves quickly in the beginning generations, but it gets stuck and converges to a solution far from the global optimum. It can be seen that ERA tends to work in an exploration manner throughout the evolution, where s is on the minimum value while a and b are around the maximum values, as shown in Fig. 27b. Meanwhile, Fig. 27c shows that ERA evolves slowly and converges to a solution near the global optimum. It works dynamically in exploration and exploitation strategy during the evolution, where s is fluctuating between the minimum and the medium values, but a and b tends on the maximum values, as shown in Fig. 27d.

Fig. 27



Convergence and adaptation curves for the global trajectory optimization [Instruction: "problem" --> "problems"]problem of Cassini1 and Messenger.

The convergence and adaptation curves above prove that the proposed adaptation scheme effectively controls the exploration and exploitation balance. This scheme makes ERA can handle many types of benchmark functions: unimodal, multimodal, shifted, rotated, and also real-world problems.

4 Conclusions

The proposed evolutionary Rao algorithm (ERA) works very well based on two additional schemes: splitting the population into two subpopulations based on their qualities: high and low, with a proper portion adaptively during the evolution, and exploiting two evolutionary operators: crossover and mutation. The evaluations of twenty-three classic benchmark functions and ten CEC-C06 2019 benchmarks show that it significantly outperforms all the competitors: Rao-1, Rao-2, Rao-3, and FA-CL, where it reaches the Friedman mean rank of 1.52 and 1.50, respectively, with the p-values of Wilcoxon rank-sum test of less than 0.05 for most of the benchmarks. Examining the five real-world global trajectory optimization problems inform that ERA gives significant performances only for some of the competitors. Detailed investigations prove that all the proposed schemes work well as they are designed and make ERA effectively control the exploration and exploitation balance. All the proposed schemes make ERA able to handle most of the benchmark functions with various types: unimodal, multimodal, shifted, rotated, and also real-world problems. However, in the future, a new advanced adaptation scheme to update the population size dynamically throughout the evolutionary process as well as a better mutation scheme will be created to improve the performance of ERA. Besides, it will be comprehensively examined using more challenging benchmarks.

Declaration of Competing Interest

The authors report no declarations of interest.

References

The corrections made in this section will be reviewed and approved by a journal production editor. The newly added/removed references and its citations will be reordered and rearranged by the production team.

[1] Hussain K., Mohd Salleh M.N., Cheng S., Shi Y., Metaheuristic research: a comprehensive survey, Artificial Intelligence Review. Intel. Rev.Artif. Intel. Rev. 52 (4) (2019) 2191–2233, doi:10.1007/s10462-017-9605-z.

[2] Holland J.H., Genetic algorithms, Scientific American. Am.Sci. Am. 267 (1) (1992) 66–72, doi:10.1038/scientificamerican0792-66.

[3] Lambora A., Gupta K., Chopra K., Genetic Algorithm- A Literature Review, in: 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon). doi:10.1109/COMITCon.2019.8862255, 2019, pp. 380–384.

[4] Murillo F., Neuenschwander T., Dornberger R., Hanne T., Optimization of a Robotic Manipulation Path by an Evolution Strategy and Particle Swarm Optimization, in: ACM International Conference Proceeding Series, Association for Computing Machinery. doi:10.1145/3396474.3396488, 2020, pp. 36–41.

[5] Li Z., Zhang Q., Variable metric evolution strategies by mutation matrix adaptation, Information Sciences. Sci.Inform. Sci. 541 (2020) 136–151, doi:10.1016/j.ins.2020.05.091.

[6] Cheng J., Pan Z., Liang H., Gao Z., Gao J., Differential evolution algorithm with fitness and diversity ranking-based mutation operator, Swarm and Evolutionary Computation 61 (2021) 100816, doi:0.1016/j.swevo.2020.10081610.1016/j.swevo.2020.100816.

[7] R. Zhang, Z. Qiu,;1; Optimizing hyper-parameters of neural networks with swarm intelligence: A novel framework for credit scoring, PLoS ONE 15 (6). doi:10.1371/journal.pone.0234254.R. Zhang, Z. Qiu, Optimizing hyper-parameters of neural networks with swarm intelligence: A novel framework for credit scoring, PLoS ONE 15 (6). doi:10.1371/journal.pone.0234254.

[8] Mp V., Anand B., Microprocessors and Microsystems Particle swarm optimization technique for multilevel inverters in solar harvesting micro grid system, Microprocessors and Microsystems 79 (August)2020103288doi:. Microsyst.Microprocess. Microsyst. 79 (2020) 103288, doi:0.1016/j.micpro.2020.10328810.1016/j.micpro.2020.103288.

[9] Kumar D., Gandhi B.G.R., Bhattacharjya R.K., Firefly Algorithm and Its Applications in Engineering OptimizationModeling and Optimization in Science and Technologiesalgorithm and its applications in engineering optimizationFirefly algorithm and its applications in engineering optimization, Model. Optimiz. Sci. Technol. 16 (2020) 93–103, doi:10.1007/978-3-030-26458-1_6.

[10] V. Kumar, D. Kumar,;1; A Systematic Review on Firefly Algorithm: Past, Present, and Future, Archives of Computational Methods in Engineeringdoi:10.1007/s11831-020-09498-y. URL https://www.x-mol.com/paper/1311351373626052608.

[11] Gupta S., Deep K., Enhanced leadership-inspired grey wolf optimizer for global optimization problems, Engineering with Computers. Comput.Eng. Comput. 36 (4) (2020) 1777–1800, doi:10.1007/s00366-019-00795-0.

[12] J. C. Bansal, S. Singh,;1; A better exploration strategy in Grey Wolf Optimizer, Journal of Ambient Intelligence and Humanized Computingdoi:10.1007/s12652-020-02153-1.

[13] Assiri A.S., Hussien A.G., Amin M., Ant lion optimization: Variants, hybrids, and applicationsAnt lion optimization: variants, hybrids, and applications, IEEE Access 8 (2020) 77746–77764, doi:10.1109/ACCESS.2020.2990338.

[14] Holland J.H., Holland J.H., Others, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence, University of Michigan press, 1975.

[15] Rizal A.C., Suyanto S., Human-Like Constrained-Mating to Make Genetic Algorithm More Explorative, in: 2020 8th International Conference on Information and Communication Technology (ICoICT), IEEE, doi:https://doi.org/10.1109/ICoICT49345.2020.9166387. URL https://ieeexplore.ieee.org/document/9166387/, 2020, pp. 1–5.

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

[17] Li D., Guo W., Lerch A., Li Y., Wang L., Wu Q., An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization, Swarm and Evolutionary Computation 60 (2021) 100789, doi:~~0.1016/j.swevo.2020.100789~~10.1016/j.swevo.2020.100789.

[18] Sedighizadeh D., Masehian E., Sedighizadeh M., Akbaripour H.[Instruction: Please update the title "ScienceDirect GEPSO: A new generalized particle swarm optimization algorithm" --> "GEPSO: A new generalized particle swarm optimization algorithm"], ScienceDirect GEPSO: A new generalized particle swarm optimization algorithm, ~~Mathematics and Computers in Simulation. Comput. Simul.~~Math. Comput. Simul. 179 (2021) 194–212, doi:10.1016/j.matcom.2020.08.013.

[19] Yang X.-S., Firefly algorithms for multimodal optimization, in: Stochastic Algorithms: Foundations and Applications, Springer Berlin Heidelberg, 2009, pp. 169–178.

[20] Fister I., Fister I., Yang X.-S., Brest J., A comprehensive review of firefly algorithms, ~~Swarm and Evolutionary ComputationEvolut. Comput.~~Swarm Evolut. Comput. 13 (2013) 34–46.

[21] Peng H., Zhu W., Deng C., Wu Z., Enhancing firefly algorithm with courtship learning, ~~Information Sciences. Sci.~~Inform. Sci. 543 (2020) 18–42, doi:10.1016/j.ins.2020.05.111.

[22] Wu S., Wu Z., Peng H., Enhancing Firefly Algorithm with Best Neighbor Guided Search Strategy, ~~Wuhan University Journal of Natural Sciences. J. Natural Sci.~~Wuhan Univ. J. Natural Sci. 24 (6) (2019) 524–536, doi:10.1007/s11859-019-1432-4.

[23] Mirjalili S., Mirjalili S.M., Lewis A., Grey Wolf Optimizer, ~~Advances in Engineering Software6920144661doi:. Eng. Softw.~~Adv. Eng. Softw. 69 (2014) 46–61, doi:~~0.1016/j.advengsoft.2013.12.007~~10.1016/j.advengsoft.2013.12.007.

[24] Luo J., Liu Z., Novel grey wolf optimization based on modified differential evolution for numerical function optimization, ~~Applied Intelligence. Intel.~~Appl. Intel. 50 (2) (2020) 468–486, doi:10.1007/s10489-019-01521-5.

[25] Bahuguna S., Pal A., Annealed grey wolf optimization, ~~Advances in Mathematics: Scientific Journal. Math.: Sci. J.~~Adv. Math.: Sci. J. 9 (8) (2020) 5477–5489, doi:10.37418/amsj.9.8.18.

[26] Long W., Wu T., Cai S., Liang X., Jiao J., Xu M., A Novel Grey Wolf Optimizer Algorithm with Refraction Learning, IEEE Access 7 (2019) 57805–57819, doi:10.1109/ACCESS.2019.2910813.

[27] M. H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, An improved grey wolf optimizer for solving engineering problems, Expert Systems with Applications 166. doi:10.1016/j.eswa.2020.113917. URL https://www.sciencedirect.com/science/article/pii/S0957417420307107.

[28] Mirjalili S., ~~Advances in Engineering Software The Ant Lion OptimizerAdvances in Engineering Softwareengineering software the ant lion optimizer~~[Instruction: Please update the title "Advances in engineering software the ant lion optimizer" --> "The ant lion optimizer"]Advances in engineering software the ant lion optimizer, Adv. Eng. Softw. 83 (2015) 80–98, doi:10.1016/j.advengsoft.2015.01.010.

[29] Rao R.V., Rao algorithms: Three metaphor-less simple algorithms for solving optimization problems, ~~International Journal of Industrial Engineering Computations. J. Ind. Eng. Comput.~~Int. J. Ind. Eng. Comput. 11 (1) (2020) 107–130 doi:10.5267/j.ijiec.2019.6.002.

[30] Abdullah J.M., ~~Fitness Dependent Optimizer: Inspired by the Bee Swarming Reproductive Process~~Fitness dependent optimizer: inspired by the bee swarming reproductive process, IEEE Access 7 (2019) 43473–43486, doi:10.1109/ACCESS.2019.2907012.

[31] Vinko T., Izzo D., Global Optimisation Heuristics and Test Problems for Preliminary Spacecraft Trajectory Design, Tech. Rep. GOHTPPSTD, 2008 URL http://www.esa.int/gsp/ACT/doc/INF/pub/ACT-TNT-INF-2008-GOHTPPSTD.pdf.

[32] Xia X., Gui L., Zhang Y., Xu X., Yu F., A fitness-based adaptive differential evolution algorithm, ~~Information Sciences5492021116141doi:10.1016/j.ins.2020.11.015. URL https://doi.org/10.1016/j.ins.2020.11.015Xia X., Gui L., Zhang Y., Xu X., Yu F., A fitness-based adaptive differential evolution algorithm, Information Sciences, 2021, 549, 116141, doi:. Sci.~~Inform. Sci. 549 (2021) 116–141, doi:~~0.1016/j.ins.2020.11.015~~10.1016/j.ins.2020.11.015.

[33] E. S. A. Advanced Concepts Team,;1; Global Trajectory Optimisation Problems Database (2021). URL https://www.esa.int/gsp/ACT/projects/gtop/.

[34] Vinko T., Izzo D., Bombardelli C., Benchmarking different global optimisation techniques for preliminary spacetrajectory design, in: 58th International Astronautical Congress, URL http://www.esa.int/gsp/ACT/doc/MAD/pub/ACT-RPR-MAD-2007-BenchmarkingDifferentGlobalOptimisationTechniques.pdf, 2007.

Biography

[Instruction: **The photo is wrong. Please update the photo with the attached file.**][Instruction: **Please change "Suyanto" --> "Suyanto Suyanto" to make it consistent with the author name below the title of the manuscript.**]**Suyanto** received the B.Sc. on Informatics Engineering from STT Telkom (now Telkom University), Bandung, Indonesia in 1998, the M.Sc. on Complex Adaptive Systems from Chalmers University of Technology, Goteborg, Sweden, in 2006, and the Doctor on Computer Science from Universitas Gadjah Mada in 2016. Since 2000, he joined STT Telkom as a lecturer in School of Computing. His research interests include artificial intelligence, machine learning, deep learning, swarm intelligence, speech processing, and computational linguistics. Scopus ID: 56843751100, Researcher ID: AAB-5223-2021, Publon ID: 4171399, Orcid: https://orcid.org/0000-0002-8897-8091.

[Instruction: **The photo is wrong. Please update the photo with the attached file.**]**Agung Toto Wibowo** received the B.Sc. on Informatics Engineering from STT Telkom (now Telkom University), Bandung, Indonesia in 2005, the M.T. on Electrical Engineering from Bandung Institute of Technology, Indonesia, in 2009, and the Doctor of Philosophy on Computing Science from University of Aberdeen in 2019. Since 2006, he joined IT Telkom as a lecturer in School of Computing. His research interests include artificial intelligence, machine learning, deep learning, swarm intelligence, speech processing, and recommender systems. Scopus ID: 57195616672

[Instruction: **The photo is wrong. Please update the photo with the attached file.**]**Said Al Faraby** received M.Sc. in Artificial Intelligence from University of Amsterdam in 2015. Before that, he completed B.Sc. on Informatics Engineering from IT Telkom (now Telkom University), Bandung, Indonesia in 2010. Since 2015 he joined Telkom University as a lecturer at School of Computing. His research interests mainly in the area of machine learning, natural language processing, and intelligence systems. Scopus ID: 55845346500, Publon ID: 4214736.

[Instruction: **The photo is wrong. Please update the photo with the attached file.**]**Siti Saadah** received the Bachelor and Master degree in Informatics Engineering from Telkom Institute of Technology (now Telkom University), Bandung, Indonesia in 2009 and 2012. Since 2009, she joined Telkom University as a lecturer in School of Computing. She is Teaching Design and Analysis Algorithm, Artificial Intelligence, Theory Authomata at Telkom University. Her research interests include machine learning, financial computing, AI healthcare, prediction and simulation. Scopus ID: 55523371300, Researcher ID: AAD-6187-2021, Publon ID: 4215578.

[Instruction: **The photo is wrong. Please update the photo with the attached file.**]**Rita Rismala** received the Bachelor and Master degree in Informatics Engineering from Telkom Institute of Technology (now Telkom University), Bandung, Indonesia in 2010 and 2013. She is now a Ph.D student in School of Electrical Engineering and Informatics, Bandung Institute of Technology, Bandung, Indonesia. Since 2011, she joined Telkom University as a lecturer in School of Computing. Her research interests include machine learning, deep learning, recommender system, and natural language processing. Scopus ID: 55844928300, Researcher ID: AAE-4960-2021, Publon ID: 4187049.

Footnotes

Article Footnotes

[★] This research is funded by the Directorate of Research and Community Service or *Direktorat Penelitian dan Pengabdian Masyarakat* (PPM), Telkom University, with grant number: 444/PNLT3/PPM/2020.

Highlights:

- An evolutionary Rao algorithm (ERA) is proposed to enhance the three state-of-the-art metaheuristic Rao algorithms by introducing two new schemes.
- The population is split into two subpopulations: high- and low-quality individuals to control searching strategy.
- Two evolutionary operators: crossover and mutation operators are incorporated to give the exploitation and exploration strategies.
- A fitness-based adaptation procedure is introduced to dynamically tune the three sensitive parameters to balance the exploitation and exploration.
- Comprehensive examinations are performed using 38 benchmarks: 23 classic, 10 CEC-C06 2019, and 5 global trajectory optimization problems.

Queries and Answers

Q1

Query: The author names have been tagged as given names and surnames (surnames are highlighted in teal color). Please confirm if they have been identified correctly.

Answer: Yes