Applied Soft Computing 107 (2021) 107389

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Population management in metaheuristic algorithms: Could less be more?

Bernardo Morales-Castañeda *, Daniel Zaldívar, Erik Cuevas, Alma Rodríguez, Mario A. Navarro

Departamento de Electrónica, Universidad de Guadalajara, CUCEI. Blvd. Marcelino García Barragán 1421, 44430, Guadalajara, Mexico

ARTICLE INFO

Article history:
Received 16 June 2020
Received in revised form 24 March 2021
Accepted 31 March 2021
Available online 9 April 2021

Keywords:
Exploration and exploitation
Balance
Population diversity
Differential evolution
Metaheuristics

ABSTRACT

In this paper, a new parameter for population management is implemented in the Differential Evolution algorithm. The new scheme integrates a set of operators that analyze the exploration and exploitation effects during its operation. With these operators, the new method obtains important knowledge about its population diversity during its evolution. Under such conditions, the proposed method can reduce its population when the diversity is too low in order to reduce its computational cost and improve its search capacities simultaneously. To test the new additions, the proposed algorithm has been tested in a set of 29 complex functions and two interplanetary trajectory design problems. The outcome of the tests demonstrates a greatly improved performance when compared to the original Differential Evolution algorithm, a few of its most successful variants, and other algorithms.

1. Introduction

Since their inception, metaheuristic algorithms have been a fixture of the optimization field [1]. These optimization methods make use of stochastic schemes to find a heuristic that could provide the solution to a certain problem. Metaheuristic algorithms are mainly utilized when dealing with complex and nonlinear optimization problems.

Most metaheuristic algorithms employ search agents that follow certain rules or behaviors to explore a feasible solution space and find possible solutions. This confers them several benefits, among them is the ability to exchange knowledge obtained between the search agents, or the development of a healthy diversity between the individuals, which ensures the efficient exploration of the solution space [2,3].

The most common classification for metaheuristic algorithms considers three different categories: physics, evolution, and swarm based, each according to the phenomenon from which each algorithm is inspired. Physics based algorithms such as the Simulated Annealing (SA) [4], the Gravitational Search Algorithm (GSA) [5] and the States of Matter Search (SMS) [6] deal with themes such as thermodynamics, gravitation and the states of matter, while instead, evolution based algorithms like the Genetic Algorithm (GA) [7], the Differential Evolution (DE) [8], the Self-Adaptative Differential Evolution (JADE) [9] or the Evolutionary Strategies (ES) [10] imitate concepts from natural evolution like reproduction and mutation. Lastly, swarm based algorithms usually find inspiration in the collective intelligence of different species of animals such as the Particle Swarm Optimization (PSO) algorithm [11], the Artificial Bee Colony (ABC) algorithm [12], the Firefly Algorithm (FA) [13], the Cuckoo Search (CS) algorithm [14], the Social Spider Optimization (SSO) algorithm [15], the Crow Search Algorithm (CSA) [16], Moth Flame Optimization (MFO) algorithm [17] and the Bat Algorithm (BA) [18].

However, there seems to be a lack of information, or knowledge, regarding how and why they enjoy such great success in the optimization field [19]. Consequently, several attempts have been conducted to analyze and understand the inner workings happening during the optimization process of any metaheuristic algorithm [20–24].

One major hurdle for the optimal performance of metaheuristic algorithms is the correct setting of their parameters [25,26]. While there are some algorithms that employ sophisticated self-adaptative parameters that reduce the need for the user to manually tune them (CMA-ES [27], SHADE [28], L-SHADE [29], MPADE [30]), most do not enjoy this capability. Many techniques have been developed to take on this problem, and they have been divided in two classifications, off-line and on-line tuning [31–34]. Off-line tuning is when the parameters are determined before running the metaheuristic algorithm, conversely, on-line tuning is when the parameters are controlled or updated during the runtime of the algorithm. Some of the most popular off-line tuning

* Corresponding author.
E-mail addresses: jb.moralescastaneda@gmail.com (B. Morales-Castañeda), daniel.zaldivar@cucei.udg.mx (D. Zaldívar), erik.cuevas@cucei.udg.mx (E. Cuevas), alma.rvazquez@academicos.udg.mx (A. Rodríguez), marioa.navarro@alumno.udg.mx (M.A. Navarro).

https://doi.org/10.1016/j.asoc.2021.107389
1568-4946/

methods are: F-Race [35], REVAC [36], ParamILS [37], SPO [38], SMAC [39], while some of the best know examples of successful on-line tuning are the previously mentioned self-adaptative algorithms, or the Adaptative Operator Selection [40] methodology which tries to dynamically find the best evolutionary operator among many, to improve the results of a metaheuristic algorithm.

In this paper, observations made in previous analysis of the exploration and exploitation balance employed by metaheuristic algorithms [20], have been used to design new search operators to increase efficiency and performance in metaheuristic algorithms. These new search operators have been added to the original Differential Evolution algorithm. This is one of the most popular optimization approaches thanks to its robustness, precision, adaptability, simplicity, and performance. It is also one of the most extensively studied metaheuristic algorithms [41]. The simplicity, adaptability, and huge amount of previous research to consult made this algorithm the perfect base in which to add the newly developed operators.

To test the effectiveness of the proposed algorithm, an experimental comparison has been conducted. The set of optimization problems utilized for this comparison is composed of 29 benchmark functions, which are comprised of an assortment of multimodal, unimodal, hybrid and shifted functions. In addition to this, the comparison contains the experimental results on two interplanetary trajectory design problems. The results demonstrate a greatly improved performance when compared to the original Differential Evolution algorithm, a few of its most successful variants, and other metaheuristic algorithms.

This paper has been organized in the following manner: Section 2, details the particulars of the Differential Evolution (DE) algorithm. In Section 3, we discuss previous work from whence the observations that became the new search operators were noted. In Section 4, the Population Reduction Differential Evolution (PRDE) is explained. Section 5, the experimental analysis of optimization functions is presented. In Section 6, the Population Reduction operator is utilized in three other algorithms. Section 7 presents an analysis of the results obtained in real-world engineering problems. Finally, Section 8, details the conclusions drawn.

2. Differential evolution

The Differential evolution is a metaheuristic algorithm for optimization proposed by Storn and Price [8] in 1997. It is one of the most popular optimization algorithms of the many which find inspiration in the natural laws of evolution.

Like most of the other evolutionary algorithms, the DE initiates its iterative process by generating a random population of search agents spread inside the boundaries of the search space of any given optimization problem. This initial population is then "evolved" through the iterative process by applying several processes such as *mutation*, *crossover*, and *selection*.

The *mutation* process involves the generation of new "mutated" solutions for each individual solution in the population. Considering the population of solutions as $\boldsymbol{X} = \{\boldsymbol{X_1}, \boldsymbol{X_2}, \ldots \boldsymbol{X_{np}}\}$ where np is the size of the population, the *mutation* process can be illustrated as:

$$\boldsymbol{M_i} = \boldsymbol{X_c} + F(\boldsymbol{X_a} - \boldsymbol{X_b}) \tag{1}$$

where a, b and c are three different population indexes chosen at random, and the parameter F is the scaling factor, which is used to control the magnitude of the differential variation $(\boldsymbol{X_a} - \boldsymbol{X_b})$. Eq. (1) represents the most basic mutation strategy, also called "rand/1", but many different variants have been proposed; the following five are some of the most popular.

"rand/2"

$$\boldsymbol{M_i} = \boldsymbol{X_f} + F\,(\boldsymbol{X_a} - \boldsymbol{X_b}) + F(\boldsymbol{X_c} - \boldsymbol{X_d}) \tag{2}$$

"best/1"

$$\boldsymbol{M_i} = \boldsymbol{X_{best}} + F(\boldsymbol{X_a} - \boldsymbol{X_b}) \tag{3}$$

"best/2"

$$\boldsymbol{M_i} = \boldsymbol{X_{best}} + F(\boldsymbol{X_a} - \boldsymbol{X_b}) + F(\boldsymbol{X_c} - \boldsymbol{X_d}) \tag{4}$$

"current to best/1"

$$\boldsymbol{M_i} = \boldsymbol{X_i} + F(\boldsymbol{X_{best}} - \boldsymbol{X_a}) + F(\boldsymbol{X_b} - \boldsymbol{X_c}) \tag{5}$$

"current to pbest/1"

$$\boldsymbol{M_i} = \boldsymbol{X_i} + F(\boldsymbol{X_p} - \boldsymbol{X_i}) + F(\boldsymbol{X_a} - \boldsymbol{X_b}) \tag{6}$$

For the previous equations a, b, c, d, and f represent five different population indexes chosen at random, $\boldsymbol{X_{best}}$ is the best individual in the current population and p is a random population index chosen from the top P individuals.

The different mutation strategies employ different tradeoffs between exploration and exploitation, each of them with its own strengths. The first two, DE/rand/1 and DE/rand/2 are completely random in nature, which confers them a strong explorative capability but with diminished convergence speed. On the other hand, DE/best/1, DE/best/2, DE/current-to-best/1 and DE/current-to-pbest/1 exploit the current best solution, which grants them an increased exploitative capability but can also lead to premature convergence. In simpler terms, the first two mutation strategies focus on exploration, while the latter four focus in exploitation [42–45].

The next is the *crossover* process; this is the combination of each solution with its "mutated" counterpart. Considering that each solution is composed of n dimensions, such that $\boldsymbol{X_i} = \{x_1, x_2, \ldots, x_n\}$, and $\boldsymbol{M_i} = \{m_1, m_2, \ldots, m_n\}$, then the combination is accomplished as follows:

$$c_j = \begin{cases} x_j & if\ rand\,(0,1) \le C_r \\ m_j & otherwise \end{cases} \quad \text{for } j = 1, \ldots, n \tag{7}$$

where C_r is the crossover rate, which is used to control the probability that an element from the combined solution was taken from the original or from the mutated solution.

Finally, in the *selection* process, the combined solution is compared to the original solution in terms of their fitness, with the best of the two being admitted to the next generation, where the whole process starts anew.

3. Literature review

While most researchers agree that metaheuristic algorithms perform better when the exploration and exploitation phases reach a correct balance [2,3,46], there are very few studies about how to reach this balance or about how balance really affects the performance of an algorithm [19]. In previous work [20], a metric known as dimension-wise diversity measurement [21,23] was utilized to quantify the level of exploration and exploitation through the optimization process of many metaheuristic algorithms; then, the balance employed was compared and contrasted with the performance achieved by each algorithm. This metric is explained in Eqs. (8) and (9).

The results from that work showed that most of the best performing algorithms employed a balance of over 90% exploitation and less than 10% exploration in all the types of optimization problems analyzed in the work. Some of the results from that work are shown in Fig. 1.

One of the conclusions gathered from [20] was that the algorithms that generally obtained the best results were spending

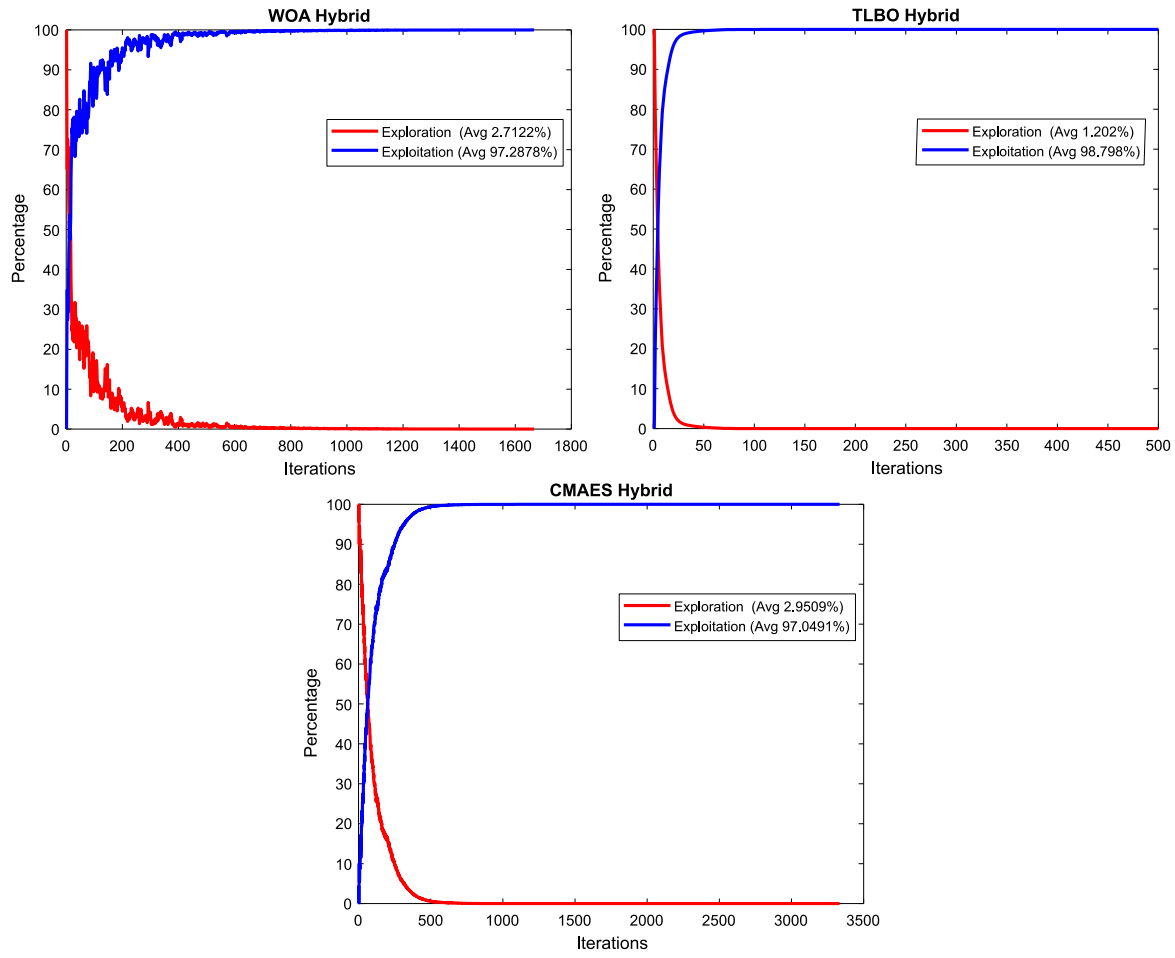

Fig. 1. Exploration and exploitation balance of top-performing metaheuristic algorithms in hybrid functions.

most of the time in the exploitation of the search space. Fig. 1 displays the exploration and exploitation balance in hybrid functions. Starting almost immediately, the exploration percentage, which translates directly to the diversity value between search agents, begins to decrease fast until it is so low that by 1/3 of the runtime, the algorithms populations were located in a space so small that the diversity between the search agents was almost null.

By pondering in this newfound knowledge, and since algorithms usually are run a finite amount of function evaluations, one hypothesis was conceived: If the population of an algorithm reaches a diversity between its search agents small enough to be almost null, at that moment, could it be reduced to one search agent and perform the same but using fewer function evaluations? And if this is true, can't those extra function evaluations be used to run the algorithm longer and potentially obtain a better performance in the quality of the solution?

Previous researchers have experimented with dynamic populations in metaheuristic algorithms, employing increasingly complex approaches. Some authors utilize a linear equation to reduce the population progressively during runtime [47,48], while others, a probability mass function to adjust the population [49], there is also work that focuses on increasing the population amount to try to improve performance [50], and algorithms that also utilize the diversity metric to adjust population, but with different methodologies/goals, like [51] which uses it as a form of diversity maintenance, increasing or reducing the population to maintain a relative balance in diversity. Finally, there are even metaheuristic algorithms that use a different external metaheuristic algorithm just to balance its population [52].

We wanted to propose a minimalistic approach to population reduction, with a different goal: improving the performance of the optimization process by reducing the population to improve the efficient use of function evaluations, and only when the diversity reaches a sufficiently small number that warrants the reduction.

4. Population reduction differential evolution

To test the hypothesis, a metaheuristic algorithm was modified. The selected algorithm was Differential Evolution with the mutation strategy "DE/rand/1". We wanted to choose one of the most computationally simple algorithms to prove that the proposed operators could help even the most basic algorithms reach and surpass the performance of the more complex recent optimization techniques. The modification consisted of a dynamic population that was reduced when a small enough diversity value was reached. A flow diagram of the Population Reduction Differential Evolution (PR-DE) can be seen inf Fig. 2.

As can be seen in Fig. 2, the additions to the classic DE algorithm take the form of two new processes: Diversity calculation and population reduction.

4.1. Diversity calculation

There are two concepts deeply integrated with every Metaheuristic algorithm; those are the concepts of exploration and exploitation. The former is a term used when the group of search

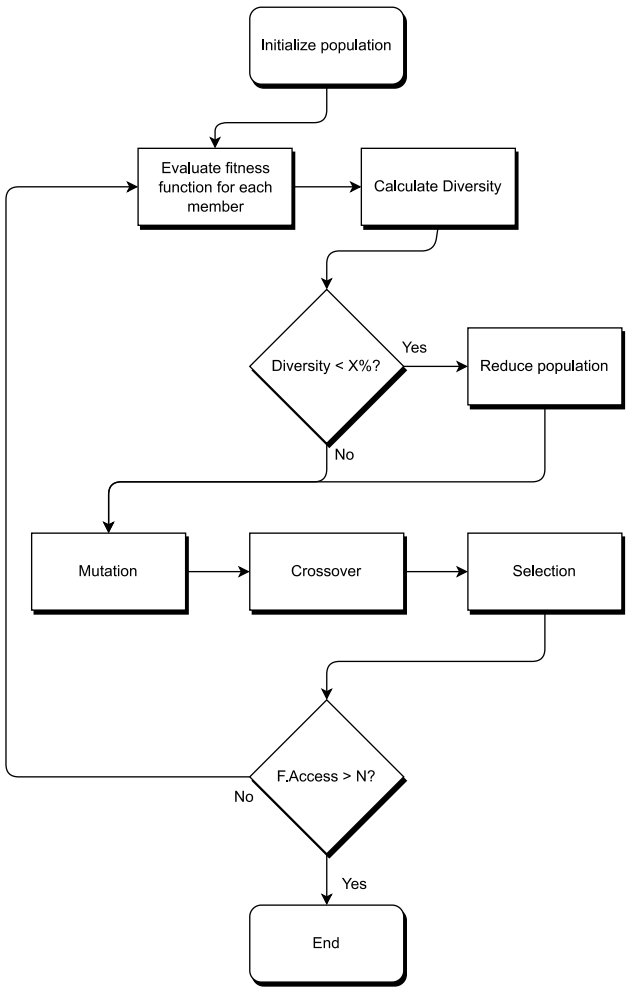

Fig. 2. Flow diagram of dynamic population differential evolution.

agents that the algorithms use to probe the search space or population are distant between one another, searching many different solutions from multiple zones. The latter, when the population is focusing on a small zone where the best solutions have been found, and because of this, the population tends to be very close together.

The dimension-wise diversity [23] is utilized to calculate this variation on the distance between the search agents. This is useful to obtain information regarding the amount of time spent exploiting or exploring the search space during the runtime of any metaheuristic algorithm. This metric defines the diversity between the search agents as:

$$Div_j = \frac{1}{n}\sum_{i=1}^{n}\left|median\left(x^j\right) - x_i^j\right| \tag{8}$$

$$Div = \frac{1}{m}\sum_{j=1}^{m} Div_j \tag{9}$$

where x_i^j is the position of the search agent i in dimension j. $median(x^j)$ is the median position of the whole population in dimension j. n corresponds to the number of search agents and m is the number of dimensions, or design variables, of the optimization problem.

In other words, Div_j is the measurement of the length between the position of each search agent in a specific dimension to the median position of the entire population in that dimension, averaged. Then this value is averaged for all the dimensions to obtain the diversity of the population Div. Both are calculated in every iteration.

This computation is added to the DE algorithm at the beginning of every iteration. This allows the algorithm to precisely know the distance between the search agents at any given moment. After every computation of the diversity calculation process, a check is done to see if the diversity value is small enough to warrant the reduction of the population. The check is formulated as:

if $Div > maxDiv$
  $maxDiv \leftarrow Div$
else if $Div < targetDiv$
  execute Population reduction

where *maxDiv* is the biggest diversity reached during the current runtime, and *targetDiv* is the user set parameter of when to employ the population reduction process.

As the diversity may vary too much between different algorithms and different optimization problems, the parameter *targetDiv* would need to be set to different values specific to each. To solve this, the value is changed to a percentage of the maximum diversity:

$$percentDiv = Div \times \frac{100\%}{maxDiv} \tag{10}$$

After this, *targetDiv* is then renamed *targetPercent*. This change ensures that no matter what the diversity looks like inside the algorithm, the population reduction will be carried out at the same percentage among all the different algorithms and optimization problems. Taking into the account these changes, the check is reformulated as:

if $Div > maxDiv$
  $maxDiv \leftarrow Div$
else if $percentDiv < targetPercent$
  execute Population reduction

The *targetPercent* parameter is paramount to the performance of the PR-DE algorithm; it determines the amount of time the algorithm will spend utilizing the full amount of search agents for exploration. If set too high, the algorithm will reduce its population prematurely, causing it to not perform as efficiently during exploration; if set too low, the algorithm may never reach the point where the diversity is low enough to initiate the population reduction. We recommend a value of 2 for multimodal, unimodal, shifted and hybrid optimization functions, and as high as 50 for real-world optimization problems where the difficulty is much greater.

4.2. Population reduction

To ensure that the best solutions are kept after the reduction, the population is sorted according to the quality of their solutions; then, the best is saved as the new reduced population. The number of search agents maintained for this new population is determined by another user set parameter: *newPopulationAmount*.

It is very important that the algorithm can run for the same amount of function evaluations, no matter the size of the population. To guarantee it, the stop criterion of the algorithm is changed to a number of function evaluations instead of a number of iterations. This allows the algorithm to use the extra function evaluations saved by reducing the population, on extra iterations of the algorithm.

After the population is reduced, the diversity check is disabled, and the algorithm continues normally for the rest of the runtime.

The *newPopulationAmount* is significant to the performance of the algorithm, but less so that the *targetPercent* parameter. The

newPopulationAmount determines how many search agents will be available for exploitation. From our tests, we suggest a value of 10. We did not have to change this value with the increase of difficulty that real-world optimization problems entail; 10 were still enough search agents to locate a precise solution.

5. Experiments

To analyze the effectiveness of the hypothesis, an experimental test comparing the results of 10 metaheuristic algorithms: Artificial Bee Colony (ABC) [12], Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [27], Crow Search Algorithm (CSA) [16], Differential Evolution (DE) [8], Self-adaptive differential evolution (JADE) [9], Adaptive Differential Evolution with Linear decrease in population size (L-SHADE) [29], Firefly Algorithm (FA) [13], Moth Flame Optimization (MFO) [17], Adaptative Differential Evolution with novel mutation strategies in multiple sub-populations (MPADE) [30], Particle Search Optimization (PSO) [11], and the proposed algorithm, in a set of 29 benchmark test functions (detailed in Appendix A). The Differential Evolution and its variants (JADE, L-SHADE, MPADE) were selected to compare directly with the original algorithm and some of its state-of-the-art variants, one of which also reduces the population but for different reasons and in a different manner. The rest were chosen to cover a wide variety of design methodologies. The availability of source code has also been an important factor. The maximum function evaluations for every algorithm were set to 5000 * Dimension to ensure a fair comparison. Boundary checks are done in every algorithm where applicable (see Table 1).

For the comparison, each algorithm has been employed to optimize a set of 29 benchmark functions. 30 independent runs were considered and from these results the Average, Median and Standard Deviation were calculated. The Average (**AB**) and Median (**MD**) allow us to evaluate the precision of the solutions, meanwhile, the Standard Deviation (**SD**) is an indicator of the dispersion of the solutions. To validate the performance differences among the algorithms, and validate if a significant statistical difference exist between them, two non-parametric tests have been considered: the Wilcoxon's Rank Sum test for independent samples [53] and, the Friedman test [54].

The experiments have been divided in the following subsections: multimodal functions in Section 5.1, unimodal in Section 5.2, then hybrid and shifted functions in Sections 5.3 and 5.4. Lastly, in Section 5.5 the results of the convergence and diversity analysis are presented. Most of the functions have been taken from literature or from [58].

5.1. Results of multimodal test functions

Multimodal functions contain multiple local optima that hinder the algorithms' ability to reach the global optimum without getting trapped. Detailed information of each function can be found in Appendix A [A.1]. Tables 2 and 3 present the optimization results of the different algorithms, considering 30 and 50 dimensions. The best results are highlighted in boldface.

Table 2 shows that in most of the 14 functions the PR-DE algorithm delivered better results than not only the original DE, or the more advanced variants but also than the other metaheuristic algorithm in the comparison. A few exceptions to this exist in functions f_7, f_{10}, f_{11} and f_{12} where the best performing algorithms were DE in the first of those functions, MPADE in the second and LSHADE in the last two. The first case is curious because the base DE algorithm managed to obtain the best results in that specific function. This was caused by a very difficult optimization function, where the improved exploitative capabilities were a detriment. DE, thanks to its rand/1 mutation strategy, focused solely on exploration and managed to escape the multiple local minimums to get closer to the global solution. For the other functions, we believe the results are attributed to the design of the PR-DE algorithm. It is basically an improvement to the exploitative capabilities of an algorithm; this means that if the original algorithms exploration cannot reach the area where the global minimum is located, the new additions cannot improve much.

Table 1
Parameter settings for each algorithm.

Algorithm	Parameters
PR-DE	*initialPopulation* = 50, *targetPercent* = 2, *newPopulationAmount* = 10, DE parameters were set the same as the original DE algorithm parameters listed below.
ABC	Colony population = 124, Onlooker bees = 62, Employed bees = 62, Scout bee = 1, limit = 100 [12].
CMA-ES	The parameters were set following the indications given by its author [55].
CSA	Crow population = 50, Flight length = 2, awareness probability = 0.1 [16].
DE	Population = 50, weight parameter $F = 0.75$, crossover probability $CR = 0.2$ [56]. The mutation strategy is DE/rand/1.
JADE	$uF = 0.6$, $uCR = 0.5$ [9].
LSHADE	Historical memory size $H = 6$, p value = 0.11, external archive size $\|\boldsymbol{A}\| = N_{pop} * 2.6$ where N_{pop} denotes the population size [29].
MPADE	The parameters were set following the indications given by its author [30].
FA	Randomness factor set to $\alpha = 0.98^{it}$, light absorption coefficient set to $\gamma = 1.0$ [13].
MFO	$N_{flames} = round\left(\left(N_{pop} - k\right) * \frac{N_{pop} - 1}{k_{max}}\right)$ [17].
PSO	Learning factors c_1 and $c_2 = 2$. Inertia weight factor decreases linearly starting at 0.9 and ending at 0.2 [57].

The experiments were also replicated, considering 50 dimensions instead of 30. By having almost double the number of design variables, the optimization problems become more difficult to solve; this is done to test how the algorithms perform in increasing challenges. The results of these tests are registered in Table 3. From the results presented, it is apparent that the proposed algorithm maintained its performance, obtaining once again the better results in the comparison, in 9 of the 14 functions.

To analyze the difference between the distinct optimization methods, and establish if a statistically significant difference between them exists, the Wilcoxon's Rank Sum test for independent samples [53], was utilized. In this test, the null hypothesis establishes that the difference between two algorithms is not discernible. Opposite that, the alternative hypothesis means that the difference between the algorithms is significant. If the p-value is smaller than the significance level of 0.05 the null hypothesis is rejected. The resultant p-values are reported in Table 4. They have been obtained from the results of multimodal functions with 30 design variables. To aid in the analysis of the following table, the symbols ▲, ▼ and ► are utilized. These symbols indicate if the PR-DE algorithm had better performance, less performance or equal performance than the compared method. From the p-values shown below, it is evident that all of them are below the significance level. This indicates that the proposed method is significantly distinct from the other methods. The few NaN values were there in the cases when both algorithms managed to reach the global solution in all the runs.

Table 2
Results for multimodal functions with 30 design variables.

Function		PR-DE	DE	JADE	LSHADE	MPADE	ABC	CMA-ES	CSA	FA	MFO	PSO
f_1	AB	**5.22E−15**	6.82E−09	1.64E+00	1.46E−14	6.63E−15	4.88E+00	1.30E+00	1.72E+01	3.56E−01	6.08E+00	5.73E−09
	MD	**7.11E−15**	6.18E−09	1.65E+00	1.42E−14	**7.11E−15**	4.90E+00	4.20E−09	1.73E+01	3.56E−01	7.14E−09	3.95E−09
	SD	**6.49E−16**	2.26E−09	4.79E−01	4.70E−15	1.23E−15	9.22E−01	4.95E+00	3.93E−01	3.20E−02	8.47E+00	6.09E−09
f_2	AB	**6.67E−01**	**6.67E−01**	**6.67E−01**	**6.67E−01**	**6.67E−01**	8.91E−01	**6.67E−01**	1.51E+05	8.78E−01	1.82E+04	**8.04E−01**
	MD	**6.67E−01**	**6.67E−01**	**6.67E−01**	**6.67E−01**	**6.67E−01**	7.37E−01	**6.67E−01**	1.53E+05	8.24E−01	8.38E+01	**6.67E−01**
	SD	1.06E−06	1.45E−03	1.27E−05	0.00E+00	**0.00E+00**	2.94E−01	7.29E−06	3.59E+04	1.12E−01	6.39E+04	4.82E−01
f_3	AB	**2.73E−179**	2.71E−39	2.88E−31	6.82E−79	9.03E−90	1.31E−18	1.16E−57	1.23E−01	3.07E−14	6.85E−36	3.94E−31
	MD	**1.52E−183**	1.63E−39	8.24E−38	7.35E−84	2.63E−91	1.23E−19	1.03E−58	1.27E−01	3.00E−14	5.39E−39	2.01E−32
	SD	**0.00E+00**	4.98E−39	7.52E−31	3.73E−78	3.97E−89	4.94E−18	4.56E−57	3.65E−02	1.34E−14	3.62E−35	1.34E−30
f_4	AB	**1.50E−32**	4.30E−17	3.63E−02	1.98E−30	1.03E−29	3.55E+00	5.45E+01	5.42E+01	1.26E−01	2.70E+01	3.46E−01
	MD	**1.50E−32**	4.23E−17	7.71E−09	1.01E−30	6.11E−30	2.89E+00	5.58E+01	5.53E+01	5.30E−03	2.51E+01	4.10E−18
	SD	**1.11E−47**	1.92E−17	1.16E−01	2.81E−30	1.07E−29	1.97E+00	1.28E+01	7.43E+00	2.91E−01	1.25E+01	7.03E−01
f_5	AB	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.20E−09	6.59E−16	**0.00E+00**	**0.00E+00**	8.21E−02	6.76E−10	**0.00E+00**	2.50E−31
	MD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	8.46E−02	6.38E−10	**0.00E+00**	**0.00E+00**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.20E−08	3.56E−15	**0.00E+00**	**0.00E+00**	2.08E−02	3.12E−10	**0.00E+00**	6.10E−31
f_6	AB	**0.00E+00**	7.09E−278	1.61E−77	1.12E−190	4.57E−84	2.03E−41	2.60E−54	3.71E+00	6.65E−60	**0.00E+00**	7.46E−94
	MD	**0.00E+00**	5.89E−287	3.54E−94	4.55E−231	2.75E−109	4.66E−50	1.29E−205	3.49E−01	4.76E−61	**0.00E+00**	3.23E−238
	SD	**0.00E+00**	**0.00E+00**	8.50E−77	**0.00E+00**	1.46E−83	1.11E−40	1.42E−53	1.23E+01	1.94E−59	**0.00E+00**	4.09E−93
f_7	AB	1.17E+01	**1.32E+00**	1.86E+01	3.97E+01	3.29E+01	6.91E+02	3.68E+01	6.77E+75	1.16E+02	2.15E+02	1.37E+02
	MD	3.04E+00	**5.15E−01**	3.42E+00	3.00E+01	7.97E+00	4.42E+02	3.23E+00	1.22E+74	7.87E+01	8.54E+01	5.42E+01
	SD	2.16E+01	**2.16E+00**	4.44E+01	3.92E+01	6.11E+01	9.25E+02	6.99E+01	3.33E+76	9.19E+01	2.72E+02	2.40E+02
f_8	AB	**3.00E+01**	**3.00E+01**	**3.00E+01**	**3.00E+01**	**3.00E+01**	3.28E+01	**3.00E+01**	5.74E+01	**3.00E+01**	3.32E+01	**3.00E+01**
	MD	**3.00E+01**	**3.00E+01**	**3.00E+01**	**3.00E+01**	**3.00E+01**	3.30E+01	**3.00E+01**	5.80E+01	**3.00E+01**	3.25E+01	**3.00E+01**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.65E+00	**0.00E+00**	2.08E+00	**0.00E+00**	3.94E+00	**0.00E+00**
f_9	AB	**2.08E−28**	5.64E+01	3.12E−01	3.86E−03	9.50E−14	1.07E+01	1.73E−12	1.89E+10	1.30E+03	9.13E−14	3.52E−11
	MD	**1.70E−28**	5.79E+01	2.03E−01	4.22E−20	1.07E−18	5.82E+00	1.42E−12	1.93E+10	1.31E+03	1.51E−14	8.31E−12
	SD	**8.92E−29**	1.34E+01	3.71E−01	1.26E−02	3.55E−13	1.50E+01	1.26E−12	4.83E+09	1.42E+02	2.23E−13	5.28E−11
f_{10}	AB	1.03E+01	1.03E+01	1.07E+01	8.74E+00	8.89E+00	**8.48E+00**	8.67E+00	2.40E+01	9.29E+00	1.29E+01	9.27E+00
	MD	1.03E+01	1.03E+01	1.06E+01	8.77E+00	8.97E+00	**8.51E+00**	8.82E+00	2.40E+01	9.41E+00	1.02E+01	9.21E+00
	SD	3.72E−01	3.54E−01	5.02E−01	**3.08E−01**	5.15E−01	3.61E−01	4.68E−01	2.48E+00	4.55E−01	7.06E+00	3.97E−01
f_{11}	AB	2.57E+01	2.69E+01	2.14E+01	5.11E+00	**2.85E+00**	6.57E+01	8.38E+00	2.53E+05	5.03E+01	7.10E+04	2.96E+01
	MD	2.55E+01	2.63E+01	2.45E+01	5.04E+00	**2.70E+00**	6.98E+01	8.42E+00	2.47E+05	2.78E+01	5.58E+04	2.44E+01
	SD	2.26E+00	2.24E+00	1.39E+01	1.12E+00	1.31E+00	5.50E+01	**4.02E−01**	6.52E+04	3.43E+01	6.04E+04	1.75E+01
f_{12}	AB	3.87E−01	1.20E+00	3.07E+00	2.92E−08	**8.41E−09**	4.17E+00	4.25E−07	4.91E+01	7.45E−01	4.75E+01	8.23E−01
	MD	2.95E−01	1.22E+00	2.35E−02	2.28E−08	**7.72E−09**	4.12E+00	3.98E−07	4.95E+01	5.62E−01	5.00E+01	8.10E−01
	SD	2.61E−01	1.45E−01	8.15E+00	2.41E−08	**4.54E−09**	1.11E+00	1.88E−07	3.15E+00	5.45E−01	9.76E+00	2.20E−01
f_{13}	AB	**9.46E−56**	3.17E−09	9.51E−01	3.68E−09	8.86E−04	2.61E+02	1.33E+28	1.13E+29	4.24E+00	4.50E+02	6.40E+01
	MD	**6.25E−56**	3.01E−09	8.53E−01	2.11E−11	4.45E−04	2.62E+02	1.15E−04	2.23E+28	4.35E+00	4.50E+02	1.44E−09
	SD	**1.11E−55**	9.10E−10	7.57E−01	1.76E−08	1.02E−03	6.12E+01	7.25E+28	2.26E+29	3.68E−01	2.19E+02	1.42E+02
f_{14}	AB	**−1.17E+03**	**−1.17E+03**	**−1.17E+03**	**−1.17E+03**	−1.16E+03	−9.95E+02	−9.67E+02	−7.31E+02	−1.09E+03	−1.04E+03	−1.01E+03
	MD	**−1.17E+03**	**−1.17E+03**	**−1.17E+03**	**−1.17E+03**	−1.16E+03	−9.98E+02	−9.70E+02	−7.33E+02	−1.09E+03	−1.03E+03	−1.01E+03
	SD	4.22E−14	**0.00E+00**	1.29E+01	2.58E+00	1.29E+01	3.04E+01	5.04E+01	2.86E+01	2.96E+01	3.31E+01	3.54E+01
Win		5	1	0	0	2	0	0	0	0	0	0
Draw		5	4	4	3	2	2	3	0	1	2	1
Lose		4	9	10	11	10	12	11	14	13	12	13
Average rank		**2.79**	4.54	5.50	3.89	3.64	7.96	5.68	10.93	7.11	7.64	6.32
p-value		7.48E−12										

5.2. Results of unimodal test functions

Unimodal functions contain only one optimum, they can help evaluate the accuracy, or exploitative capabilities, of the algorithms. Detailed information can be found in Appendix A [A.2]. Tables 5 and 6 presents the optimization results considering 30 and 50 dimensions. The best results are highlighted in boldface.

As previously said, the main improvement of the new search operators developed for this modified algorithm is in the exploitation capabilities. The results in Table 5 seem to confirm this. In all unimodal functions, the proposed methodology produced much more accurate solutions, with less variation in the results. The superior capabilities demonstrated with these results come from the increased iterations dedicated to the exploitation phase, gained from the eliminated surplus caused by having a huge population with the almost imperceptible difference between each search agent. To be more concise, reducing a big population to only a small amount of the best search agents liberated many function evaluations to be used in more iterations.

To test the algorithm in more difficult problems, the experiments were repeated considering 50 dimensions. Once again, the proposed method proved to be the best performing in all the functions. But more importantly, something interesting came to light. The relation between the original DE algorithm and the proposed variation can be glanced at by comparing the results in 30 and 50 dimensions. The new operators do not, in any way, modify the base algorithm; they are algorithm agnostic additions that could be possibly added to an algorithm to improve its exploitation capability. The additions only begin to take effect once the base algorithm exploration has ended, and the search agents begin to close in a zone of the search space. This means that it depends on the base algorithm to find the correct zone where the global solution is going to be found; any impairment or difficulty that the algorithm experiments while reaching that zone directly translates to fewer improvements that can be gained by the additions. This is very apparent here by looking at the original DE results in 30 and 50 dimensions. The performance worsened by the increase in the difficulty of the optimization problem.

Table 3
Results for multimodal functions with 50 design variables.

Function		PR-DE	DE	JADE	LSHADE	MPADE	ABC	CMA-ES	CSA	FA	MFO	PSO
f_1	AB	1.33E−14	4.03E−07	1.47E+00	2.53E−14	**8.41E−15**	5.54E+00	5.04E−12	1.81E+01	8.47E−02	1.59E+01	1.25E−07
	MD	1.42E−14	3.56E−07	1.40E+00	2.84E−14	**7.11E−15**	5.52E+00	3.44E−12	1.81E+01	8.42E−02	1.80E+01	6.97E−08
	SD	**1.60E−15**	2.48E−07	4.37E−01	5.00E−15	2.72E−15	8.71E−01	4.50E−12	2.80E−01	4.16E−03	5.72E+00	1.51E−07
f_2	AB	**6.67E−01**	6.68E−01	6.84E−01	**6.67E−01**	**6.67E−01**	1.11E+00	**6.67E−01**	7.98E+05	9.36E−01	2.05E+05	1.24E+00
	MD	**6.67E−01**	**6.67E−01**	**6.67E−01**	**6.67E−01**	**6.67E−01**	9.68E−01	**6.67E−01**	8.04E+05	7.81E−01	3.63E+04	**6.67E−01**
	SD	2.36E−06	2.22E−03	5.45E−02	**0.00E+00**	9.67E−17	4.88E−01	3.17E−08	1.10E+05	4.57E−01	3.17E+05	1.48E+00
f_3	AB	**1.13E−115**	1.34E−25	1.14E−29	2.61E−80	1.53E−107	6.66E−19	3.30E−71	4.95E−01	1.45E−16	1.16E−01	2.42E−24
	MD	**2.87E−119**	1.01E−25	1.84E−33	2.42E−81	2.92E−111	9.33E−20	1.28E−72	5.24E−01	1.36E−16	8.82E−32	7.32E−25
	SD	**4.25E−115**	1.35E−25	3.23E−29	7.57E−80	5.52E−107	2.05E−18	1.13E−70	1.25E−01	8.33E−17	3.54E−01	4.24E−24
f_4	AB	**1.83E−32**	1.73E−12	3.38E−01	3.30E−30	8.95E−03	4.60E+00	6.13E+01	1.31E+02	2.85E+00	5.67E+01	1.92E+00
	MD	**1.50E−32**	1.70E−12	8.95E−02	2.70E−30	1.50E−32	4.05E+00	7.44E+01	1.30E+02	2.27E+00	5.22E+01	1.73E+00
	SD	**1.82E−32**	7.27E−13	4.37E−01	2.66E−30	2.73E−02	2.92E+00	4.22E+01	1.11E+01	2.23E+00	1.65E+01	1.78E+00
f_5	AB	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.71E−07	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.71E−01	6.87E−12	**0.00E+00**	4.16E−30
	MD	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.75E−07	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.72E−01	5.76E−12	**0.00E+00**	3.16E−30
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.77E−07	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.62E−02	3.04E−12	**0.00E+00**	4.63E−30
f_6	AB	**0.00E+00**	8.72E−299	1.32E−131	**0.00E+00**	1.61E−143	3.97E−48	1.75E−236	1.49E+05	3.20E−112	**0.00E+00**	4.28E−136
	MD	**0.00E+00**	**0.00E+00**	3.99E−160	**0.00E+00**	2.19E−217	6.64E−59	**0.00E+00**	1.56E+04	4.53E−118	**0.00E+00**	1.88E−253
	SD	**0.00E+00**	**0.00E+00**	7.24E−131	**0.00E+00**	8.81E−143	2.15E−47	**0.00E+00**	3.33E+05	1.46E−111	**0.00E+00**	2.34E−135
f_7	AB	5.34E+01	**1.07E+01**	3.18E+09	2.51E+02	2.29E+02	1.00E+10	5.58E+02	1.86E+153	1.07E+91	2.40E+74	3.17E+11
	MD	3.00E+01	**7.74E+00**	5.05E+01	1.19E+02	3.00E+01	1.00E+10	1.86E+02	6.27E+151	2.24E+65	9.76E+37	1.97E+05
	SD	9.47E+01	1.07E+01	1.74E+10	3.42E+02	5.42E+02	**0.00E+00**	8.88E+02	6.55E+04	5.85E+91	1.32E+75	1.32E+12
f_8	AB	**3.00E+01**	**3.00E+01**	**3.00E+01**	**3.00E+01**	**3.00E+01**	3.66E+01	**3.00E+01**	8.52E+01	3.03E+01	4.17E+01	**3.00E+01**
	MD	**3.00E+01**	**3.00E+01**	**3.00E+01**	**3.00E+01**	**3.00E+01**	3.60E+01	**3.00E+01**	8.55E+01	3.00E+01	4.10E+01	**3.00E+01**
	SD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.69E+00	**0.00E+00**	3.22E+00	4.50E−01	7.99E+00	**0.00E+00**
f_9	AB	**5.01E−28**	2.20E+03	3.15E−01	2.02E−10	5.72E−28	2.88E+01	4.62E−18	5.45E+10	3.95E+02	6.25E+09	4.21E−08
	MD	6.87E−28	2.22E+03	1.09E−01	1.60E−22	**5.74E−28**	1.18E+01	2.68E−18	5.52E+10	3.92E+02	6.65E−12	6.99E−09
	SD	**1.20E−28**	2.30E+02	4.41E−01	1.10E−09	4.02E−28	4.08E+01	4.70E−18	7.74E+09	3.03E+01	1.91E+10	1.65E−07
f_{10}	AB	2.02E+01	2.04E+01	2.03E+01	1.67E+01	1.70E+01	**1.63E+01**	1.65E+01	7.16E+01	1.83E+01	3.80E+01	1.80E+01
	MD	2.02E+01	2.05E+01	2.02E+01	1.67E+01	1.71E+01	**1.63E+01**	1.65E+01	7.28E+01	1.85E+01	2.68E+01	1.80E+01
	SD	6.23E−01	6.05E−01	9.65E−01	**3.57E−01**	6.15E−01	4.38E−01	4.16E−01	8.16E+00	8.28E−01	2.70E+01	5.19E−01
f_{11}	AB	4.37E+01	4.62E+01	6.05E+01	2.54E+01	**1.71E+01**	1.13E+02	2.17E+01	6.81E+05	7.91E+01	2.43E+05	6.30E+01
	MD	4.25E+01	4.55E+01	5.69E+01	2.56E+01	**1.36E+01**	1.06E+02	2.17E+01	6.92E+05	8.20E+01	2.02E+05	4.51E+01
	SD	3.25E+00	2.46E+00	3.50E+01	1.19E+00	1.41E+01	7.24E+01	**5.21E−01**	1.24E+05	3.98E+01	1.68E+05	3.08E+01
f_{12}	AB	1.21E+01	1.21E+01	2.31E+01	1.02E−06	**6.18E−09**	5.69E+00	2.01E−08	6.00E+01	1.82E+01	7.80E+01	4.24E+00
	MD	1.19E+01	1.20E+01	2.21E+01	8.25E−07	**3.40E−09**	5.43E+00	1.89E−08	6.03E+01	1.81E+01	7.88E+01	4.16E+00
	SD	1.09E+00	1.19E+00	1.70E+01	9.54E−07	**6.73E−09**	1.22E+00	8.58E−09	2.41E+00	4.35E+00	5.60E+00	6.76E−01
f_{13}	AB	**1.34E−48**	6.89E−07	1.21E+00	9.58E−12	3.74E−10	1.93E+04	3.02E−06	2.34E+53	2.16E+00	9.57E+02	2.49E+02
	MD	**5.17E−49**	6.66E−07	1.13E+00	6.66E−12	3.29E−12	4.25E+02	1.26E−07	3.31E+52	2.18E+00	9.50E+02	3.39E−08
	SD	**2.70E−48**	2.13E−07	9.76E−01	8.61E−12	1.69E−09	1.03E+05	1.05E−05	9.77E+53	1.09E−01	3.23E+02	4.11E+02
f_{14}	AB	**−1.96E+03**	**−1.96E+03**	−1.91E+03	**−1.96E+03**	−1.86E+03	−1.64E+03	−1.61E+03	−1.12E+03	−1.75E+03	−1.72E+03	−1.69E+03
	MD	**−1.96E+03**	**−1.96E+03**	−1.96E+03	**−1.96E+03**	−1.87E+03	−1.64E+03	−1.62E+03	−1.12E+03	−1.75E+03	−1.73E+03	−1.69E+03
	SD	**2.43E−13**	5.43E−09	7.94E+01	6.14E+00	4.41E+01	5.34E+01	4.70E+01	3.93E+01	4.67E+01	7.16E+01	4.20E+01
Win		4	1	0	0	3	1	0	0	0	0	0
Draw		5	3	2	4	3	1	3	0	0	2	1
Lose		5	10	12	10	8	12	11	14	14	12	13
Average rank		**2.86**	4.89	6.07	3.39	3.04	7.43	4.46	10.93	7.79	8.64	6.50
p-value		1.69E−14										

Table 4
Wilcoxon test for multimodal functions with 30 design variables.

Function	PR-DE vs. DE	PR-DE vs. JADE	PR-DE vs. LSHADE	PR-DE vs. MPADE	PR-DE vs. ABC	PR-DE vs. CMAES	PR-DE vs. CSA	PR-DE vs. FA	PR-DE vs. MFO	PR-DE vs. PSO
f_1	1.72E−12▲	1.72E−12▲	8.85E−10▲	2.53E−02▼	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲
f_2	3.87E−11►	1.16E−02►	4.28E−12►	4.28E−12►	2.86E−11▲	9.33E−10►	2.86E−11▲	2.86E−11▲	3.37E−08▲	9.53E−01▲
f_3	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲
f_4	1.21E−12▲	1.21E−12▲	1.65E−11▲	1.21E−12▲	1.21E−12▲	1.21E−12▲	1.21E−12▲	1.21E−12▲	1.21E−12▲	1.21E−12▲
f_5	NaN►	NaN►	3.34E−01▲	6.61E−05▲	NaN►	NaN►	1.21E−12▲	1.21E−12▲	NaN►	3.08E−04▲
f_6	1.21E−12▲	1.21E−12▲	1.21E−12▲	1.21E−12▲	1.21E−12▲	1.21E−12▲	1.21E−12▲	1.21E−12▲	NaN►	6.25E−10▲
f_7	5.55E−02▼	4.04E−01▲	6.57E−02▲	7.62E−01▲	3.20E−09▲	7.73E−02▲	3.02E−11▲	8.10E−10▲	2.20E−07▲	7.20E−05▲
f_8	NaN►	NaN►	NaN►	NaN►	3.89E−12▲	NaN►	1.06E−12▲	NaN►	1.19E−05▲	NaN►
f_9	2.75E−11▲	2.75E−11▲	2.75E−11▲	2.75E−11▲	2.75E−11▲	2.75E−11▲	2.75E−11▲	2.75E−11▲	2.75E−11▲	2.75E−11▲
f_{10}	7.28E−01►	1.68E−04►	3.02E−11▼	3.69E−11▼	3.02E−11▼	3.02E−11▼	3.02E−11▲	9.76E−10▼	9.35E−01▲	8.89E−10▼
f_{11}	1.76E−02►	3.03E−02▼	3.02E−11▼	3.02E−11▼	3.64E−02▲	3.02E−11▼	3.02E−11▲	3.83E−05▲	3.26E−07▲	1.30E−01▲
f_{12}	5.49E−11▲	1.64E−05▲	3.02E−11▼	3.02E−11▼	3.02E−11▲	3.02E−11▼	3.02E−11▲	1.78E−04▲	3.02E−11▲	2.38E−07▲
f_{13}	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	2.99E−11▲	3.02E−11▲
f_{14}	3.34E−01►	1.72E−12►	2.08E−02►	1.30E−06►	1.72E−12▲	1.68E−12▲	1.72E−12▲	1.72E−12▲	1.69E−12▲	1.66E−12▲

Table 5
Results for unimodal functions with 30 design variables.

Function		PR-DE	DE	JADE	LSHADE	MPADE	ABC	CMA-ES	CSA	FA	MFO	PSO
f_{15}	AB	**5.69E−93**	2.31E−15	2.65E−09	8.14E−31	1.26E−26	6.26E+00	5.05E−16	1.14E+05	5.92E+00	1.79E+04	7.32E−15
	MD	**4.18E−95**	1.95E−15	8.14E−11	2.64E−31	6.55E−27	2.95E+00	3.37E−16	1.17E+05	5.94E+00	4.29E+03	1.18E−15
	SD	**2.53E−92**	1.47E−15	1.17E−08	1.26E−30	1.22E−26	7.37E+00	4.38E−16	1.65E+04	8.39E−01	2.93E+04	2.66E−14
f_{16}	AB	**3.35E−91**	5.62E−14	1.02E−06	2.80E−29	5.83E−23	1.36E+03	8.05E−15	3.97E+06	2.47E+02	1.11E+06	6.67E+02
	MD	**3.33E−92**	5.03E−14	2.74E−09	1.63E−29	4.18E−23	1.15E+03	7.42E−15	4.02E+06	2.51E+02	9.10E+05	3.40E−14
	SD	**7.20E−91**	2.76E−14	3.87E−06	3.58E−29	5.22E−23	7.32E+02	5.32E−15	4.88E+05	3.50E+01	8.86E+05	2.54E+03
f_{17}	AB	**1.29E−96**	1.10E−18	2.04E−11	3.69E−34	3.17E−34	1.05E−07	1.13E−20	4.91E+01	2.65E−03	3.50E+00	1.85E−18
	MD	**1.85E−98**	8.94E−19	7.28E−15	1.65E−34	2.41E−34	7.03E−08	7.64E−21	4.99E+01	2.70E−03	8.69E−21	3.97E−19
	SD	**6.18E−96**	6.39E−19	9.48E−11	5.41E−34	2.37E−34	1.33E−07	8.50E−21	7.44E+00	3.68E−04	9.06E+00	4.31E−18
f_{18}	AB	**1.21E−95**	4.40E−17	2.43E−11	1.68E−32	2.40E−28	1.85E−01	6.96E−19	2.43E+03	1.37E−01	5.50E+02	1.09E−16
	MD	**2.61E−97**	4.00E−17	4.07E−14	5.29E−33	1.72E−28	1.18E−01	4.70E−19	2.38E+03	1.41E−01	4.50E+02	3.30E−17
	SD	**2.45E−95**	2.04E−17	9.51E−11	2.64E−32	2.15E−28	1.74E−01	4.56E−19	2.97E+02	2.02E−02	6.58E+02	2.37E−16
f_{19}	AB	**1.11E−139**	6.93E−53	5.21E−21	1.38E−69	4.68E−65	1.67E−09	5.12E−10	7.60E−03	2.46E−09	5.07E−49	1.19E−46
	MD	**2.29E−153**	6.22E−54	2.04E−24	5.34E−73	1.54E−66	1.28E−09	4.14E−10	7.29E−03	2.29E−09	3.62E−53	7.98E−49
	SD	**6.00E−139**	2.10E−52	1.97E−20	4.86E−69	1.07E−64	1.43E−09	4.70E−10	3.78E−03	1.71E−09	2.27E−48	3.60E−46
Win		5	0	0	0	0	0	0	0	0	0	0
Draw		0	0	0	0	0	0	0	0	0	0	0
Lose		0	5	5	5	5	5	5	5	5	5	5
Average rank		**1.00**	4.80	6.80	2.20	2.80	8.80	4.80	11.00	8.40	9.00	6.40
p-value		1.50E−06										

Table 6
Results for unimodal functions with 50 design variables.

Function		PR-DE	DE	JADE	LSHADE	MPADE	ABC	CMA-ES	CSA	FA	MFO	PSO
f_{15}	AB	**3.71E−82**	1.40E−11	6.39E−07	3.00E−30	2.03E−37	1.01E+01	9.65E−21	4.17E+05	1.67E+00	1.03E+05	1.67E−12
	MD	**2.23E−83**	1.40E−11	3.62E−12	1.61E−30	4.76E−38	8.06E+00	9.43E−21	4.13E+05	1.68E+00	6.66E+04	7.67E−13
	SD	**9.79E−82**	4.97E−12	3.37E−06	3.91E−30	4.63E−37	7.79E+00	4.70E−21	3.97E+04	1.75E−01	9.70E+04	2.87E−12
f_{16}	AB	**1.39E−79**	4.81E−10	1.62E−05	1.22E−27	2.04E−31	3.06E+03	3.00E−18	2.74E+07	1.21E+02	7.90E+06	9.00E+03
	MD	**3.11E−80**	4.70E−10	4.50E−10	6.16E−28	1.02E−31	3.00E+03	2.14E−18	2.78E+07	1.20E+02	7.31E+06	3.33E−11
	SD	**2.17E−79**	1.50E−10	7.39E−05	1.88E−27	3.50E−31	1.32E+03	1.88E−18	2.51E+06	1.28E+01	4.89E+06	3.29E+04
f_{17}	AB	**6.20E−86**	5.41E−15	1.14E−13	1.39E−33	3.19E−49	2.88E−08	2.46E−26	1.09E+02	4.18E−04	2.01E+01	4.26E−16
	MD	**1.62E−87**	5.05E−15	1.07E−15	2.95E−34	8.78E−50	2.32E−08	2.19E−26	1.08E+02	4.19E−04	1.31E+01	1.81E−16
	SD	**2.75E−85**	1.84E−15	5.04E−13	3.05E−33	8.59E−49	2.18E−08	2.03E−26	8.56E+00	4.74E−05	2.25E+01	5.10E−16
f_{18}	AB	**3.28E−84**	3.66E−13	3.10E−09	5.74E−32	2.05E−39	3.34E−01	3.78E−24	9.58E+03	3.91E−02	2.70E+03	5.53E−14
	MD	**7.49E−85**	3.65E−13	4.02E−14	3.64E−32	9.84E−40	3.12E−01	2.67E−24	9.52E+03	3.80E−02	1.95E+03	1.28E−14
	SD	**7.77E−84**	1.23E−13	1.52E−08	5.51E−32	2.87E−39	2.39E−01	3.98E−24	9.81E+02	3.92E−03	2.37E+03	1.41E−13
f_{19}	AB	5.26E−40	6.12E−40	4.87E−20	1.77E−75	**5.16E−94**	2.86E−10	8.35E−11	8.51E−03	4.51E−10	7.61E−44	6.49E−46
	MD	1.48E−40	2.05E−40	6.06E−25	1.96E−78	**7.36E−99**	1.52E−10	4.96E−11	8.33E−03	3.70E−10	7.75E−48	1.66E−48
	SD	1.12E−39	1.23E−39	2.33E−19	8.75E−75	**2.56E−93**	3.42E−10	8.23E−11	4.82E−03	3.09E−10	3.17E−43	2.81E−45
Win		4	0	0	0	1	0	0	0	0	0	0
Draw		0	0	0	0	0	0	0	0	0	0	0
Lose		1	4	4	4	4	4	4	4	4	4	4
Average rank		**1.80**	5.80	6.80	2.80	1.80	8.60	4.80	11.00	8.40	8.80	5.40
p-value		6.27E−06										

This translated directly to worse performance on the proposed algorithm, because the base DE algorithm had trouble reaching the point where the additions could begin to work.

The Table 7 contains the results of the Wilcoxon test for unimodal functions considering 30 dimensions. These results are well below the 5% significance level. It can be said with statistical certainty that the algorithm is significantly different from the other optimization methods.

5.3. Results of hybrid test functions

Hybrid functions are multimodal formulations produced from several multimodal functions. This confers them with highly complex behaviors. Detailed information can be found in Appendix A [A.3]. Tables 8 and 9 show the optimization results considering 30 and 50 dimensions. The best results are highlighted in boldface.

When taking into consideration both tables, it is easily seen that PR-DE managed to obtain better performance in two of the functions. For function f_{21} in 30 dimensions, DE, LSHADE, MPADE and CMA-ES reached similar results, but the increase in difficulty from 30 to 50 dimensions affected MPADE greatly. In 50 dimensions, it could not compete with the other top-performing algorithms. The PR-DE results are thanks to the combination of a successful exploration strategy of the original DE algorithm, and the improved exploitation capabilities obtained by the proposed modifications.

The results of running the Wilcoxon test on the outcomes of the hybrid functions experimental comparison are presented in Table 10. Once again, the PR-DE algorithm, compared to every other algorithm, resulted in *p*-values below the 5% significance value, confirming the statistical difference between the distinct methodologies.

5.4. Results of shifted test functions

Having tested the performance of the algorithm when dealing with unimodal, multimodal, and hybrid optimization problems, we look at shifted functions. These functions are designed so the location of their global optimum is shifted toward different locations. These are used to test if any possible bias of the different optimization methods toward functions with the global optimum in the origin position (0,0). More detailed information

Table 7
Wilcoxon test for unimodal functions with 30 design variables.

Function	PR-DE vs. DE	PR-DE vs. JADE	PR-DE vs. LSHADE	PR-DE vs. MPADE	PR-DE vs. ABC	PR-DE vs. CMAES	PR-DE vs. CSA	PR-DE vs. FA	PR-DE vs. MFO	PR-DE vs. PSO
f_{15}	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.01E−11▲	3.02E−11▲
f_{16}	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲
f_{17}	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.01E−11▲	3.02E−11▲
f_{18}	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	2.98E−11▲	3.02E−11▲
f_{19}	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲

Table 8
Results for hybrid functions with 30 design variables.

Function		PR-DE	DE	JADE	LSHADE	MPADE	ABC	CMA-ES	CSA	FA	MFO	PSO
f_{20}	AB	**3.39E−57**	4.52E−10	4.82E−02	6.37E−15	1.18E−14	7.62E−03	3.10E+03	2.09E+04	1.93E+00	2.58E+04	1.25E−10
	MD	**1.22E−57**	4.34E−10	1.37E−05	2.50E−15	1.07E−14	7.41E−03	1.36E−07	2.05E+04	1.90E+00	3.01E+04	7.31E−11
	SD	**4.85E−57**	1.28E−10	1.82E−01	8.65E−15	7.82E−15	2.96E−03	1.01E+04	2.90E+03	2.27E−01	2.04E+04	1.55E−10
f_{21}	AB	**2.90E+01**	**2.90E+01**	2.91E+01	**2.90E+01**	2.96E+01	1.61E+02	**2.90E+01**	7.16E+02	7.29E+01	1.42E+02	5.38E+01
	MD	**2.90E+01**	**2.90E+01**	2.91E+01	**2.90E+01**	2.90E+01	1.62E+02	**2.90E+01**	7.23E+02	7.43E+01	1.02E+02	5.11E+01
	SD	**1.13E−14**	6.33E−14	1.12E−01	1.36E−14	2.17E+00	3.42E+01	1.95E−14	6.38E+01	2.02E+01	1.04E+02	1.50E+01
f_{22}	AB	**2.90E+01**	**2.90E+01**	2.93E+01	**2.90E+01**	**2.90E+01**	2.07E+02	2.24E+02	8.83E+02	4.52E+01	8.35E+02	3.88E+01
	MD	**2.90E+01**	**2.90E+01**	2.92E+01	**2.90E+01**	**2.90E+01**	2.10E+02	2.90E+01	8.81E+02	4.29E+01	6.52E+02	2.90E+01
	SD	**3.39E−57**	4.52E−10	4.82E−02	6.37E−15	1.18E−14	7.62E−03	3.10E+03	2.09E+04	1.93E+00	2.58E+04	1.25E−10
Win		1	0	0	0	0	0	0	0	0	0	0
Draw		2	2	0	2	1	0	1	0	0	0	0
Lose		0	1	3	1	2	3	2	3	3	3	3
Average rank		**2.00**	3.33	5.67	2.33	3.83	8.00	6.83	10.67	7.67	10.00	5.67
p-value		6.63E−03										

Table 9
Results for hybrid functions with 50 design variables.

Function		PR-DE	DE	JADE	LSHADE	MPADE	ABC	CMA-ES	CSA	FA	MFO	PSO
f_{20}	AB	**3.87E−61**	1.90E−09	7.30E−02	1.04E−16	5.04E−16	6.97E−03	1.43E−09	6.90E+04	2.22E−01	6.42E+04	2.29E−10
	MD	**1.66E−61**	1.79E−09	3.95E−05	7.93E−17	1.26E−17	6.86E−03	7.01E−12	5.47E+04	2.21E−01	5.02E+04	7.17E−11
	SD	**6.24E−61**	3.55E−10	2.84E−01	1.01E−16	1.61E−15	2.72E−03	7.73E−09	5.22E+04	1.55E−02	3.61E+04	4.18E−10
f_{21}	AB	**4.90E+01**	**4.90E+01**	4.91E+01	**4.90E+01**	1.38E+02	2.62E+02	**4.90E+01**	1.47E+03	1.79E+02	4.80E+02	1.24E+02
	MD	**4.90E+01**	**4.90E+01**	4.91E+01	**4.90E+01**	1.38E+02	2.58E+02	**4.90E+01**	1.51E+03	1.85E+02	4.21E+02	1.23E+02
	SD	2.27E−14	1.39E−11	4.26E−02	**1.65E−14**	2.93E+01	5.19E+01	1.96E−14	1.04E+02	2.75E+01	3.51E+02	2.31E+01
f_{22}	AB	**4.90E+01**	**4.90E+01**	4.91E+01	**4.90E+01**	1.23E+02	3.08E+02	**4.90E+01**	3.53E+03	1.53E+02	1.99E+03	1.08E+02
	MD	**4.90E+01**	**4.90E+01**	4.91E+01	**4.90E+01**	1.25E+02	3.03E+02	**4.90E+01**	2.27E+03	1.53E+02	2.02E+03	1.11E+02
	SD	**1.48E−14**	2.22E−08	8.25E−02	3.75E−14	2.02E+01	4.82E+01	2.91E−10	3.60E+03	2.90E+01	1.05E+03	2.62E+01
Win		1	0	0	0	0	0	0	0	0	0	0
Draw		2	2	0	2	0	0	2	0	0	0	0
Lose		0	1	3	1	3	3	1	3	3	3	3
Average rank		**2.00**	3.67	6.00	2.33	5.67	8.33	3.33	11.00	8.33	10.00	5.33
p-value		3.05E−03										

Table 10
Wilcoxon test for hybrid functions with 30 design variables.

Function	PR-DE vs. DE	PR-DE vs. JADE	PR-DE vs. LSHADE	PR-DE vs. MPADE	PR-DE vs. ABC	PR-DE vs. CMAES	PR-DE vs. CSA	PR-DE vs. FA	PR-DE vs. MFO	PR-DE vs. PSO
f_{20}	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	3.02E−11▲	2.89E−11▲	3.02E−11▲
f_{21}	2.89E−11▲	2.90E−11▲	2.66E−11▲	2.86E−11▲	2.90E−11▲	2.84E−11▲	2.90E−11▲	2.90E−11▲	2.90E−11▲	2.90E−11▲
f_{22}	1.41E−11▲	1.41E−11▲	1.39E−11▲	1.41E−11▲	1.41E−11▲	1.41E−11▲	1.41E−11▲	1.41E−11▲	1.41E−11▲	1.41E−11▲

is presented in Appendix A [A.4]. Tables 11 and 12 presents the results considering 30 and 50 dimensions. The best results are highlighted in boldface.

This test is interesting not only to compare the algorithms between them, but also to compare them against themselves. Each of these shifted functions has a non-shifted version in the previous unimodal, multimodal, or hybrid tests. Some algorithms manage to obtain the same result they previously obtained in non-shifted functions; others do not. The PR-DE algorithm not only obtained better performance than the other algorithms in both 30 and 50 dimensions, but it was also impervious to the shift in the functions. This imperviousness to shifts in optimization problems comes from the original DE algorithm, and it is one of the reasons it was chosen as a base for the proposed additions.

Table 13 contains the results of the Wilcoxon test in shifted functions. Once again, the outcome of this test confirms the statistical difference between the proposed algorithms and the rest of the algorithms in this comparison.

5.5. Convergence and diversity analysis

To compare the evolution of the algorithms, in terms of the best solutions being found through the runtime of their optimization process, a convergence analysis has been conducted. The objective of this test is to assess how quickly a certain method reaches its best fitness value.

Table 11
Results for shifted functions with 30 design variables.

Function		PR-DE	DE	JADE	LSHADE	MPADE	ABC	CMA-ES	CSA	FA	MFO	PSO
f_{23}	AB	**7.22E−15**	2.97E−07	2.46E+00	7.84E−14	2.80E−13	4.76E+00	6.47E−01	1.72E+01	5.74E−01	7.24E+00	1.59E−07
	MD	**7.11E−15**	2.81E−07	2.42E+00	6.93E−14	2.63E−13	4.81E+00	4.47E−07	1.73E+01	5.79E−01	7.51E−01	1.12E−07
	SD	**6.49E−16**	7.74E−08	5.10E−01	3.72E−14	1.35E−13	7.53E−01	3.54E+00	4.44E−01	4.47E−02	8.28E+00	1.44E−07
f_{24}	AB	2.76E+01	2.97E+01	3.49E+01	9.98E+00	**8.52E+00**	1.94E+02	1.26E+01	5.13E+05	4.89E+01	8.31E+04	4.44E+01
	MD	2.72E+01	2.78E+01	2.57E+01	1.00E+01	**8.39E+00**	1.69E+02	1.27E+01	5.04E+05	2.92E+01	6.92E+04	2.51E+01
	SD	3.19E+00	5.66E+00	2.51E+01	8.54E−01	1.06E+00	1.02E+02	**5.03E−01**	1.17E+05	2.96E+01	4.60E+04	3.75E+01
f_{25}	AB	**4.71E−29**	4.84E−12	5.91E−09	8.57E−25	6.60E−21	9.23E+00	1.11E−12	1.08E+05	1.20E+01	1.73E+04	3.79E−12
	MD	**0.00E+00**	4.63E−12	2.01E−11	1.79E−25	3.53E−21	5.43E+00	9.04E−13	1.06E+05	1.22E+01	6.44E+03	8.37E−13
	SD	**2.00E−28**	1.90E−12	2.63E−08	2.10E−24	8.70E−21	9.91E+00	6.45E−13	1.30E+04	1.77E+00	2.61E+04	6.19E−12
f_{26}	AB	**2.84E−26**	1.19E−10	1.97E−05	3.36E−23	1.83E−17	1.72E+03	2.65E−11	3.97E+06	4.78E+02	6.80E+05	2.10E−10
	MD	**0.00E+00**	1.07E−10	5.44E−10	1.22E−23	1.48E−17	1.63E+03	2.10E−11	3.97E+06	4.79E+02	3.95E+05	3.43E−11
	SD	**1.03E−25**	4.37E−11	9.48E−05	7.44E−23	1.67E−17	7.78E+02	2.01E−11	5.52E+05	7.15E+01	7.75E+05	5.42E−10
f_{27}	AB	**2.37E−16**	2.97E−07	1.49E+00	3.14E−07	1.00E−01	2.60E+02	2.83E+35	1.06E+29	5.92E+00	4.03E+02	3.12E+01
	MD	**0.00E+00**	3.00E−07	1.47E+00	2.63E−08	5.91E−02	2.67E+02	1.18E−02	2.85E+28	5.96E+00	4.00E+02	5.28E−08
	SD	**1.30E−15**	6.68E−08	9.88E−01	7.72E−07	1.13E−01	6.68E+01	1.55E+36	1.82E+29	4.21E−01	2.11E+02	1.07E+02
f_{28}	AB	**4.21E−31**	2.55E−15	4.80E−12	2.29E−28	1.51E−27	2.37E−06	3.11E−17	4.78E+01	5.04E−03	2.62E+00	3.03E−15
	MD	**0.00E+00**	2.53E−15	2.96E−15	1.33E−28	1.38E−27	1.49E−06	2.60E−17	4.82E+01	5.14E−03	2.44E−17	2.58E−16
	SD	**2.30E−30**	8.90E−16	2.44E−11	3.30E−28	8.54E−28	2.43E−06	1.77E−17	6.61E+00	6.53E−04	8.00E+00	8.54E−15
f_{29}	AB	**2.02E−29**	1.05E−13	2.95E−10	1.05E−26	1.28E−22	2.23E−01	2.56E−15	2.51E+03	2.73E−01	5.30E+02	1.15E−13
	MD	**0.00E+00**	9.12E−14	2.17E−13	7.31E−27	9.79E−23	1.46E−01	2.04E−15	2.50E+03	2.75E−01	3.50E+02	4.28E−14
	SD	**6.61E−29**	4.86E−14	1.54E−09	1.04E−26	1.16E−22	2.37E−01	1.68E−15	2.13E+02	3.44E−02	5.66E+02	3.39E−13
Win		6	0	0	0	1	0	0	0	0	0	0
Draw		0	0	0	0	0	0	0	0	0	0	0
Lose		1	7	7	7	6	7	7	7	7	7	7
Average rank		**1.43**	4.71	6.71	2.14	2.86	8.43	5.29	10.86	7.86	9.86	5.86
p-value		2.15E−09										

Table 12
Results for shifted functions with 50 design variables.

Function		PR-DE	DE	JADE	LSHADE	MPADE	ABC	CMA-ES	CSA	FA	MFO	PSO
f_{23}	AB	1.39E−14	2.08E−06	1.79E+00	3.81E−14	**7.11E−15**	5.56E+00	8.31E−11	1.80E+01	1.24E−01	1.80E+01	4.89E−07
	MD	1.42E−14	1.94E−06	1.77E+00	3.91E−14	**7.11E−15**	5.63E+00	6.77E−11	1.81E+01	1.25E−01	1.89E+01	2.82E−07
	SD	1.43E−15	6.22E−07	3.69E−01	8.75E−15	**0.00E+00**	8.33E−01	6.81E−11	2.76E−01	8.13E−03	3.61E+00	6.58E−07
f_{24}	AB	4.68E+01	4.88E+01	6.27E+01	2.78E+01	**1.81E+01**	2.58E+02	2.51E+01	1.39E+06	8.86E+01	2.68E+05	6.90E+01
	MD	4.38E+01	4.69E+01	4.76E+01	2.79E+01	**1.78E+01**	2.29E+02	2.50E+01	1.40E+06	9.70E+01	2.54E+05	4.82E+01
	SD	5.21E+00	4.09E+00	3.73E+01	9.28E−01	1.73E+00	1.08E+02	**4.65E−01**	2.57E+05	4.57E+01	1.10E+05	2.91E+01
f_{25}	AB	**5.72E−29**	4.78E−10	3.87E−07	7.39E−27	3.58E−28	1.24E+01	2.04E−18	4.29E+05	3.20E+00	1.17E+05	3.30E−11
	MD	**0.00E+00**	4.44E−10	4.03E−11	5.93E−27	2.78E−28	1.11E+01	1.84E−18	4.31E+05	3.24E+00	1.25E+05	1.07E−11
	SD	**2.12E−28**	1.64E−10	1.49E−06	4.35E−27	3.38E−28	1.05E+01	1.29E−18	3.66E+04	4.70E−01	7.37E+04	5.08E−11
f_{26}	AB	**8.29E−26**	1.78E−08	1.70E−05	3.99E−24	6.63E−25	3.54E+03	7.70E−16	2.60E+07	2.31E+02	8.60E+06	1.33E+03
	MD	**0.00E+00**	1.65E−08	1.62E−09	2.80E−24	5.53E−25	3.65E+03	6.43E−16	2.68E+07	2.30E+02	5.48E+06	5.30E−10
	SD	**2.07E−25**	7.90E−09	8.70E−05	4.81E−24	4.82E−25	1.48E+03	5.69E−16	3.19E+06	3.70E+01	9.82E+06	7.30E+03
f_{27}	AB	**2.37E−16**	6.23E−06	1.40E+00	3.27E−09	1.47E−07	4.52E+02	2.44E−04	3.36E+53	2.97E+00	9.23E+02	2.48E+02
	MD	**0.00E+00**	5.78E−06	1.15E+00	4.31E−10	4.88E−09	4.46E+02	5.93E−06	2.04E+52	2.95E+00	9.50E+02	4.54E−06
	SD	**1.30E−15**	1.52E−06	1.04E+00	1.39E−08	3.31E−07	8.67E+01	9.92E−04	8.53E+53	1.65E−01	3.43E+02	4.02E+02
f_{28}	AB	**3.37E−30**	1.65E−13	2.42E−11	2.31E−29	2.95E−30	1.89E−07	5.02E−24	1.07E+02	8.19E−04	1.22E+01	9.48E−15
	MD	**0.00E+00**	1.56E−13	1.97E−16	2.52E−29	0.00E+00	1.25E−07	4.42E−24	1.09E+02	7.90E−04	1.64E−15	5.14E−15
	SD	**9.91E−30**	4.92E−14	1.31E−10	1.76E−29	5.43E−30	1.76E−07	2.76E−24	1.10E+01	8.88E−05	1.65E+01	1.40E−14
f_{29}	AB	**7.87E−29**	1.19E−11	4.19E−11	1.13E−27	3.88E−28	2.79E−01	8.46E−22	9.44E+03	7.33E−02	2.05E+03	6.49E−13
	MD	**0.00E+00**	1.20E−11	5.00E−15	9.91E−28	3.28E−28	2.01E−01	6.77E−22	9.49E+03	7.50E−02	1.45E+03	3.76E−13
	SD	**1.77E−28**	3.26E−12	2.05E−10	7.94E−28	3.72E−28	1.91E−01	6.36E−22	9.77E+02	7.54E−03	1.97E+03	9.78E−13
Win		5	0	0	0	2	0	0	0	0	0	0
Draw		0	0	0	0	0	0	0	0	0	0	0
Lose		2	7	7	7	5	7	7	7	7	7	7
Average rank		**1.71**	5.43	6.71	2.86	1.71	8.86	3.86	10.93	7.71	10.07	6.14
p-value		2.34E−10										

For this experiment, a representative set of eight functions, operated in 50 dimensions, were selected. To build the convergence graphs, the raw data generated in the experiments was utilized. Since the experiments were repeated 30 times, the convergence data selected was that of the run, which most accurately represented the median result. The following figures present the convergence graphs for the compared algorithms. Fig. 3 exhibits the convergence results for functions f_3 and f_{13}, Fig. 4 for functions f_{15} and f_{18}, Fig. 5 for functions f_{20} and f_{21}, and Fig. 6 for functions f_{25} and f_{27}. In all graphs, the x-axis represents the function evaluations, and the y-axis represents the best fitness values found.

From Figs. 3 to 6, the graphs shown clearly demonstrate the increased capabilities gained from reducing the population according to its internal diversity in order to spend more function evaluations on the exploitation phase. The convergence responses confirm the effectiveness of the proposed algorithm.

To analyze how, or if, the population diversity changes from the original DE algorithm to the modified version, the following figures show the evolution of the population diversity through

Table 13
Wilcoxon test for shifted functions with 30 design variables.

Function	PR-DE vs. DE	PR-DE vs. JADE	PR-DE vs. LSHADE	PR-DE vs. MPADE	PR-DE vs. ABC	PR-DE vs. CMAES	PR-DE vs. CSA	PR-DE vs. FA	PR-DE vs. MFO	PR-DE vs. PSO
f_{23}	1.72E−12▲	1.72E−12▲	1.70E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲
f_{24}	2.28E−01▲	1.67E−01▲	3.02E−11▼	3.02E−11▼	3.02E−11▲	3.02E−11▼	3.02E−11▲	9.03E−04▲	3.02E−11▲	9.82E−01▲
f_{25}	2.37E−12▲	2.37E−12▲	2.37E−12▲	2.37E−12▲	2.37E−12▲	2.37E−12▲	2.37E−12▲	2.37E−12▲	2.34E−12▲	2.37E−12▲
f_{26}	5.22E−12▲	5.22E−12▲	5.22E−12▲	5.22E−12▲	5.22E−12▲	5.22E−12▲	5.22E−12▲	5.22E−12▲	5.20E−12▲	5.22E−12▲
f_{27}	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲
f_{28}	1.72E−12▲	1.72E−12▲	1.90E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲	1.72E−12▲
f_{29}	3.16E−12▲	3.16E−12▲	3.16E−12▲	3.16E−12▲	3.16E−12▲	3.16E−12▲	3.16E−12▲	3.16E−12▲	3.06E−12▲	3.16E−12▲

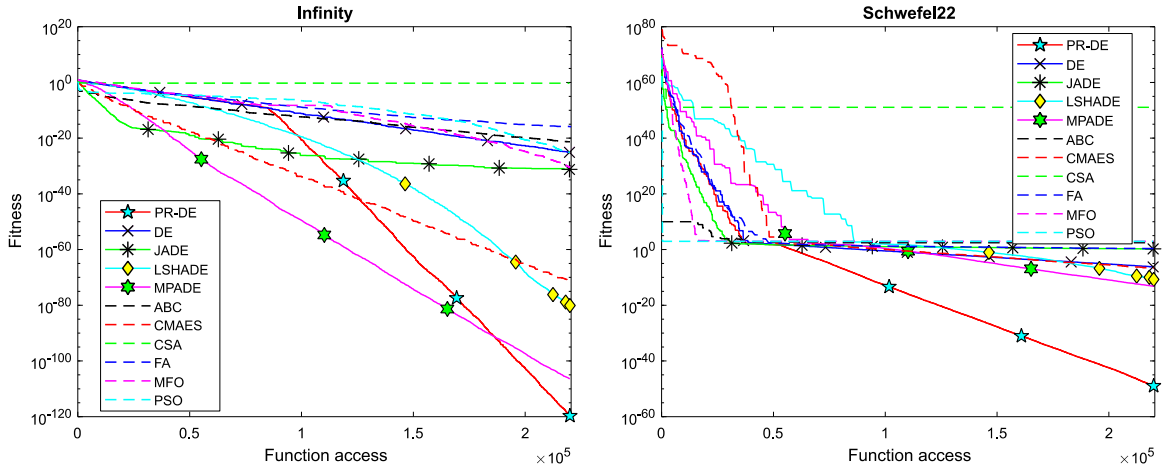

Fig. 3. Convergence curves for multimodal functions f_3 *and* f_{13}.

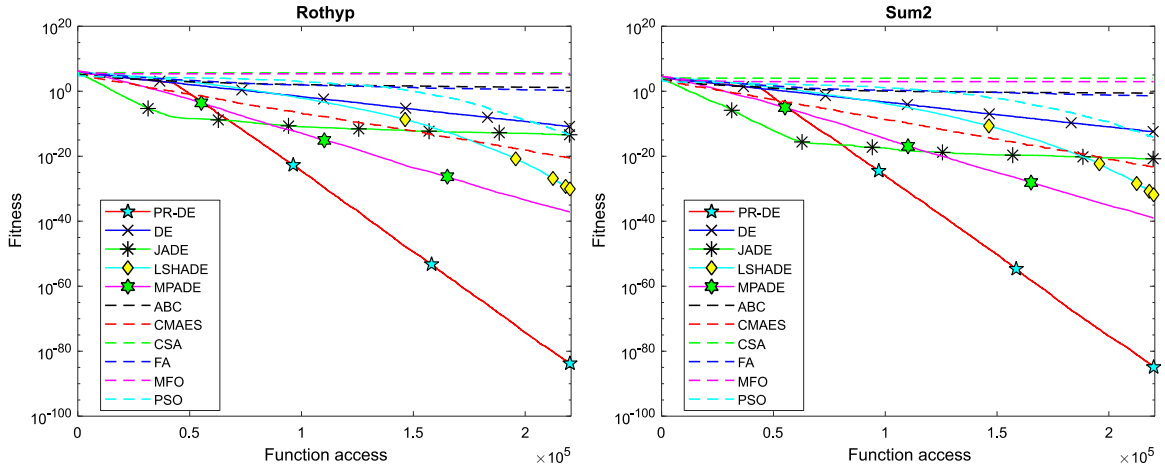

Fig. 4. Convergence curves for unimodal functions f_{15} *and* f_{18}.

the runtime of a representative set of eight functions, for the PR-DE and the DE algorithms using 250,000 function evaluations and considering 50 dimensions.

From Figs. 7 to 10, the increase in usable iterations granted by the reduction in population is very significant. From the 5000 iterations used by the DE algorithm to reach 250,000 function evaluations with a population of 50 search agents, the PR-DE algorithm manages to fit more than 20,000 iterations in the same 250,000 function evaluations. Notwithstanding this, a thorough analysis of the graphs shows that the diversity curves are almost the same, only being compressed by the increase in iterations. This can be easily seen in the Infinity function shown in Fig. 7, or the Hybrid2 function shown in Fig. 9, where both the PR-DE and the original DE algorithm follow the same curve to reach almost zero diversity at the iteration 200. This is because the Population Reduction technique only takes place after the diversity reaches a small enough value determined by the *targetPercent* parameter.

6. Population reduction in other algorithms

To see how the Population Reduction technique can affect other metaheuristic algorithms, the MPADE, CMAES and JADE algorithms were also modified. Utilizing the same representative set of functions used in the previous experiment operated in 50 dimensions, with 30 independent runs each. For the first two algorithms, the Population Reduction was configured exactly as with the PR-DE algorithm, in the other hand, for the JADE algorithm it required an adjustment in the *targetPercent* parameter, from a value of 2 to a value of 0.4, because we noted that it was

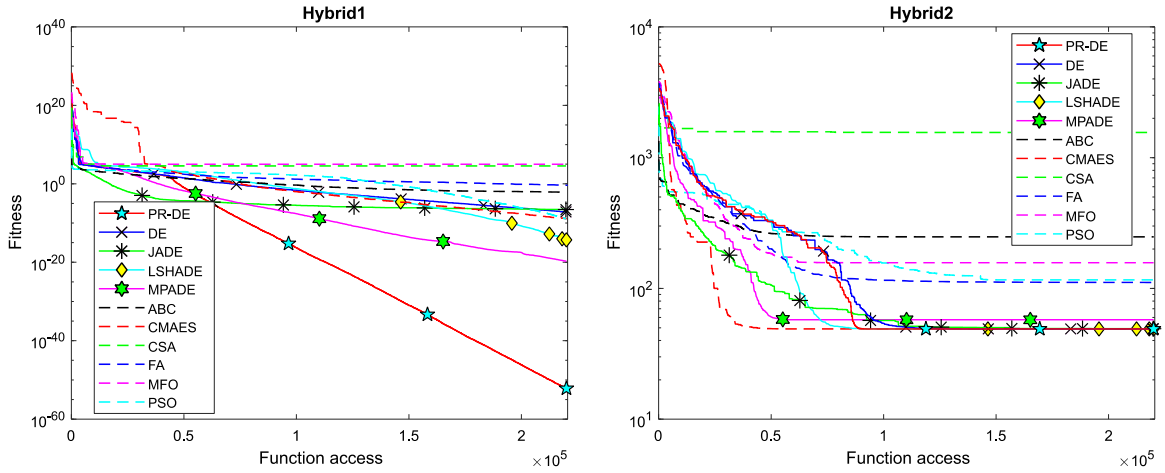

Fig. 5. Convergence curves for hybrid functions f_{20} *and* f_{21}.

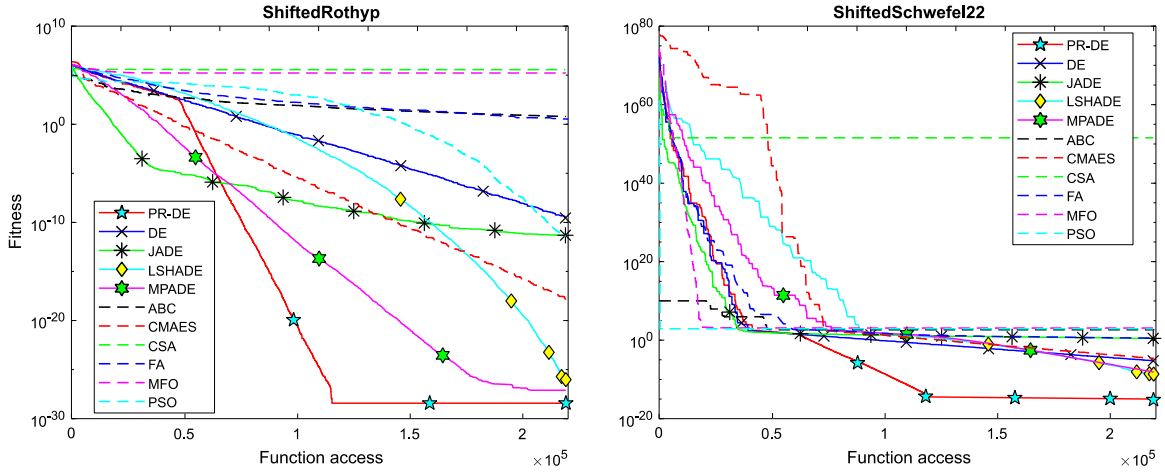

Fig. 6. Convergence curves for shifted functions f_{25} *and* f_{27}.

reducing the population too early, causing it to not reach the zone with the global minimum. The Table 14 shows the results.

This table shows a great improvement in the MPADE and CMAES algorithms, but a milder improvement in the JADE algorithm. We speculate this is caused by an inefficiency of its explorative operator, and a limitation on its exploitative capabilities. The first can be deduced by the need to reduce the *targetPercent* parameter, meaning it needs more iterations before it reaches the global minimum area compared to the other two algorithms. The second can be seen in the mild improvement in precision gained by the extra function evaluations, compared to the other two algorithms that managed to efficiently use the extra function evaluations to great results.

7. Engineering optimization problems

To assess the PR-DE optimization technique performance further, a few interplanetary trajectory design problems were solved Cassini 1 Multiple gravity assist (MGA) problems, and SAGAS Multiple gravity-assists with one deep space maneuver (MGA-1DSM) problem [59–61]. These are represented as functions f_{b1} and f_{b2} respectively. More details on both problems can be found on Appendix B.

As these are highly complex optimization problems, the difficulty increased so much that the PR-DE parameters had to be adjusted so that the population reduction actually took place during the runtime. Specifically, the *targetDiv* was set to 50, because the previously used value of 2 was never reached. The rest of the algorithms are configured as previously disclosed. The stop criterion was set to 50,000 function evaluations. The experiments are run 10 independent times.

The results are classified as ***Worst*** and ***Best-***found solution, the average best-found solutions of the 10 runs **AB**, and the standard deviation **SD**. These results can be observed in Table 15. The best solutions obtained in each interplanetary trajectory design problem are highlighted in boldface. From all the algorithms in the comparison, the proposed method managed to find the best solution to both problems. Not only that, but it also obtained the best average solution considering 10 independent runs in the two problems.

The best set of design variables found are listed in Table 16. These are the parameter configurations for the best solutions found by the PR-DE algorithm over all independent runs for the Cassini 1 MGA problem and the SAGAS MGA-1DSM problem.

8. Conclusions and future work

Understanding the amount of time an algorithm spends on exploitation, and how its population of search agents varies in diversity through the runtime, allowed us to develop a way to manage that population to improve efficiency and performance of metaheuristic algorithms. The proposed modifications to the DE algorithm, tested in a set of 29 optimization functions and

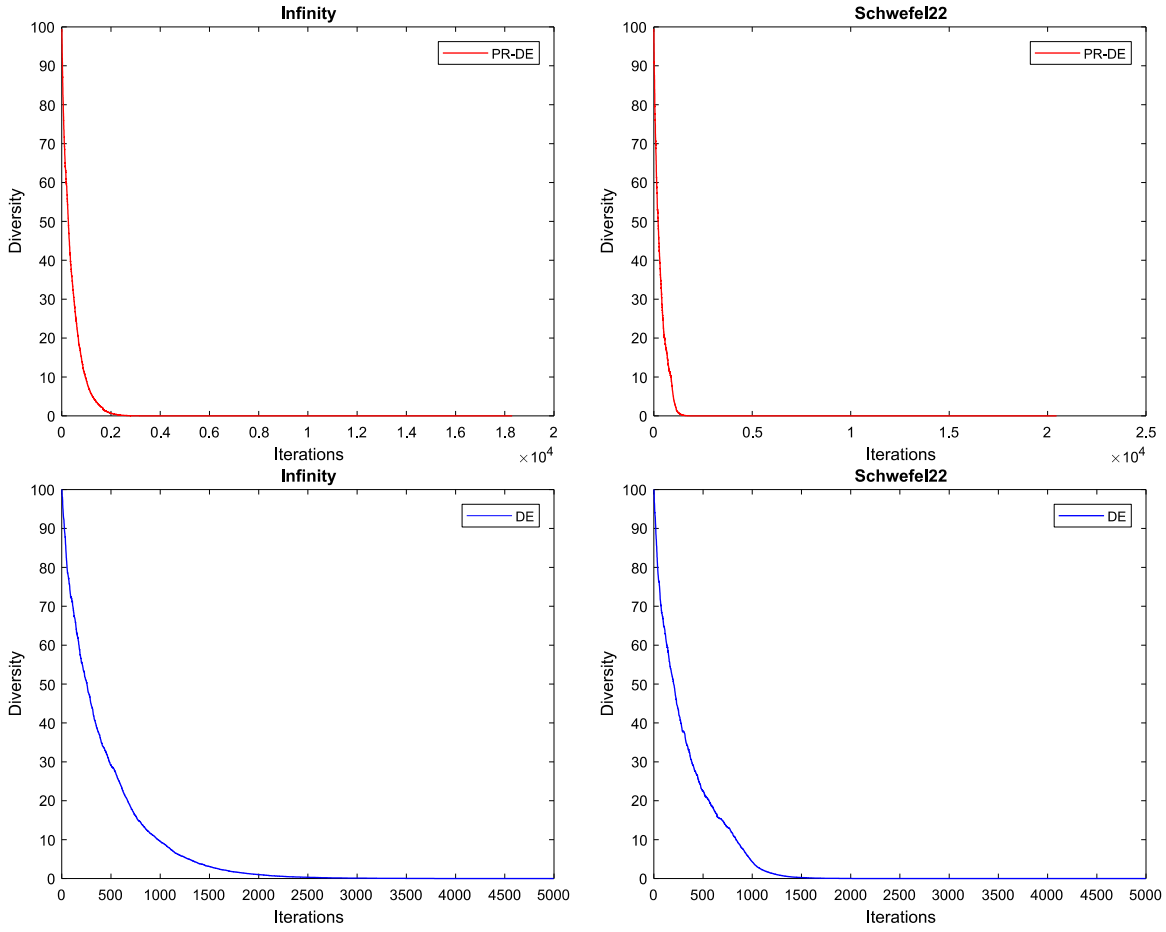

Fig. 7. Population diversity through iterations for multimodal functions f_3 *and* f_{13}.

Table 14
Comparison results for modified MPADE, CMAES and JADE.

Function		PR-MPADE	MPADE	PR-CMAES	CMAES	PR-JADE	JADE
f_3	AB	**2.43E−149**	2.72E−120	**0.00E+00**	5.70E−81	1.35E−26	**1.25E−29**
	MD	**1.47E−155**	2.17E−125	**0.00E+00**	1.14E−81	**1.67E−38**	6.61E−35
	SD	**1.33E−148**	1.49E−119	**0.00E+00**	1.11E−80	7.39E−26	**4.69E−29**
f_{13}	AB	**1.44E−27**	7.32E−13	**2.38E−22**	2.79E+38	**8.74E−01**	9.27E−01
	MD	**3.56E−34**	9.75E−16	**1.79E−26**	2.19E−09	**7.29E−01**	8.72E−01
	SD	**6.71E−27**	3.30E−12	**9.57E−22**	1.53E+39	**7.27E−01**	**7.72E−01**
f_{15}	AB	**1.09E−93**	1.13E−43	**4.42E−214**	3.61E−24	**7.39E−09**	1.02E−07
	MD	**1.66E−95**	4.37E−44	**3.17E−217**	2.82E−24	5.57E−09	**4.69E−14**
	SD	**3.59E−93**	2.74E−43	**0.00E+00**	2.66E−24	**8.59E−09**	4.61E−07
f_{18}	AB	**8.61E−94**	2.47E−45	**7.16E−278**	6.31E−28	**1.88E−10**	4.98E−10
	MD	**1.37E−96**	5.36E−46	**7.70E−279**	4.30E−28	6.05E−11	**6.31E−15**
	SD	**2.94E−93**	5.32E−45	**0.00E+00**	5.74E−28	**2.80E−10**	1.76E−09
f_{20}	AB	**6.67E−44**	5.12E−25	**2.04E−24**	2.99E+02	**7.57E−04**	2.77E−03
	MD	**2.02E−48**	3.13E−25	**3.04E−31**	1.14E−11	**1.38E−05**	5.55E−05
	SD	**3.56E−43**	7.64E−25	**1.08E−23**	1.64E+03	**2.69E−03**	8.72E−03
f_{21}	AB	**5.59E+01**	5.83E+01	6.01E+01	**4.90E+01**	**4.91E+01**	5.04E+01
	MD	**5.67E+01**	5.69E+01	5.73E+01	**4.90E+01**	**4.90E+01**	4.91E+01
	SD	**8.06E+00**	1.16E+01	9.67E+00	**2.34E−14**	**5.21E−02**	5.86E+00
f_{25}	AB	2.92E−28	**2.64E−28**	**6.62E−27**	4.20E−24	**8.96E−09**	3.69E−07
	MD	**1.14E−28**	2.08E−28	**6.53E−27**	2.75E−24	2.15E−09	**5.26E−11**
	SD	4.41E−28	**2.29E−28**	**1.53E−27**	3.96E−24	**1.48E−08**	1.77E−06
f_{27}	AB	**1.54E−15**	6.92E−13	**1.04E−13**	7.26E−06	7.56E−01	**7.32E−01**
	MD	**0.00E+00**	3.91E−14	**1.03E−13**	3.93E−09	6.80E−01	**5.09E−01**
	SD	**2.59E−15**	2.12E−12	**1.24E−14**	2.98E−05	**5.08E−01**	6.35E−01

two interplanetary trajectory design problems, demonstrated the improvement that could be gained by this knowledge.

For future work, we can see plenty of existent metaheuristic algorithms being improved by considering the management of

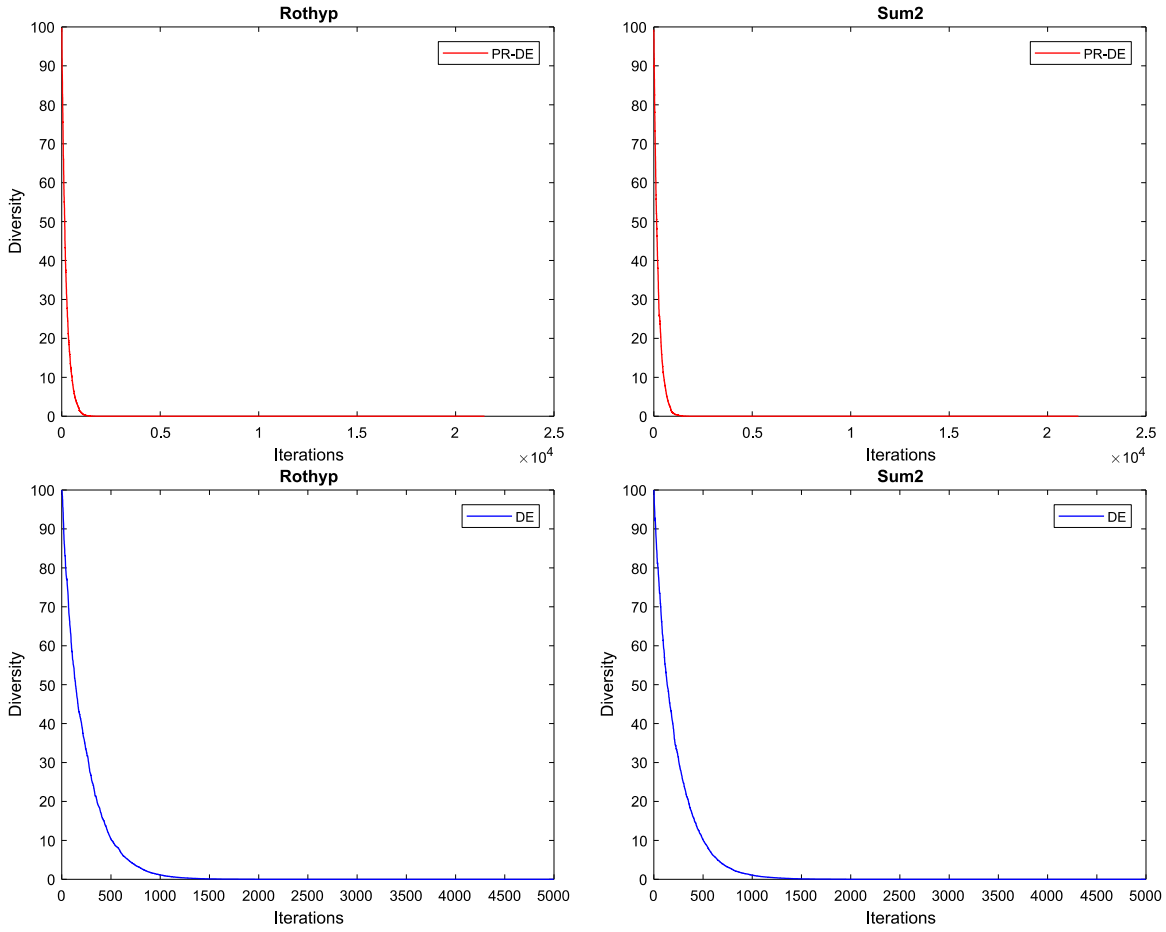

Fig. 8. Population diversity through iterations for unimodal functions f_{15} *and* f_{18}.

Table 15
Comparison results for real-world optimization problems.

Function		PR-DE	DE	JADE	LSHADE	MPADE	ABC	CMA-ES	CSA	FA	MFO	PSO
f_{b1}	*Worst*	15.6779	**12.4913**	19.7594	12.5420	19.2863	18.7378	67.2880	1.00E+300	17.4758	16.2581	17.5126
	Best	**4.9469**	6.5565	7.6197	5.3034	8.4367	11.0807	6.0736	1.00E+300	5.7383	5.8051	5.3215
	AB	**1.01E+01**	1.02E+01	1.33E+01	1.11E+01	1.30E+01	1.34E+01	2.44E+01	1.00E+300	1.07E+01	1.18E+01	1.30E+01
	SD	3.25E+00	2.46E+00	3.72E+00	**2.16E+00**	3.25E+00	2.81E+00	1.76E+01	6.55E+04	3.77E+00	2.84E+00	3.90E+00
f_{b2}	*Worst*	970.3143	965.3539	988.4494	**970.1247**	1797.5000	1438.5347	2595.1637	1.00E+300	1075.5217	1445.8037	1234.9320
	Best	**153.8810**	302.2681	614.0207	932.5844	1541.5000	938.1555	1584.7795	1.00E+300	345.8496	943.5670	376.3301
	AB	**8.19E+02**	8.28E+02	8.93E+02	9.56E+02	1.68E+03	1.08E+03	1.90E+03	1.00E+300	9.14E+02	1.04E+03	9.72E+02
	SD	2.75E+02	2.18E+02	1.44E+02	**1.85E+01**	9.10E+01	1.37E+02	3.50E+02	6.55E+04	2.03E+02	1.46E+02	2.33E+02

Table 16
Best parameter configuration for design variables obtained.

Function	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	$f_b(x)$
f_{b1}	−791.7952	159.6270	449.3858	55.1084	1014.9610	4536.5152							**4.9469**
f_{b2}	7017.0389	5.6896	1.0000	0.7011	787.1138	645.7802	0.4642	0.2355	1.0500	16.7804	−1.5818	1.5977	**153.8810**

the population presented in this paper. Another area for improvement could be a way to adaptatively set the parameters according to the evolution of the balance between exploration and exploitation during the optimization process, or a way to progressively reduce the population through the optimization process until a threshold is reached, instead of reducing it only one time. Also, further research on how metaheuristic algorithms work should be done. As already demonstrated, a better understanding on the inside mechanics that govern these optimization algorithms allows us to develop better performing methodologies.

CRediT authorship contribution statement

Bernardo Morales-Castañeda: Conceptualization, Investigation, Writing - original draft. **Daniel Zaldívar:** Resources, Data curation. **Erik Cuevas:** Resources, Writing - review & editing. **Alma Rodríguez:** Software, Validation. **Mario A. Navarro:** Visualization, Methodology.

Declaration of competing interest

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.

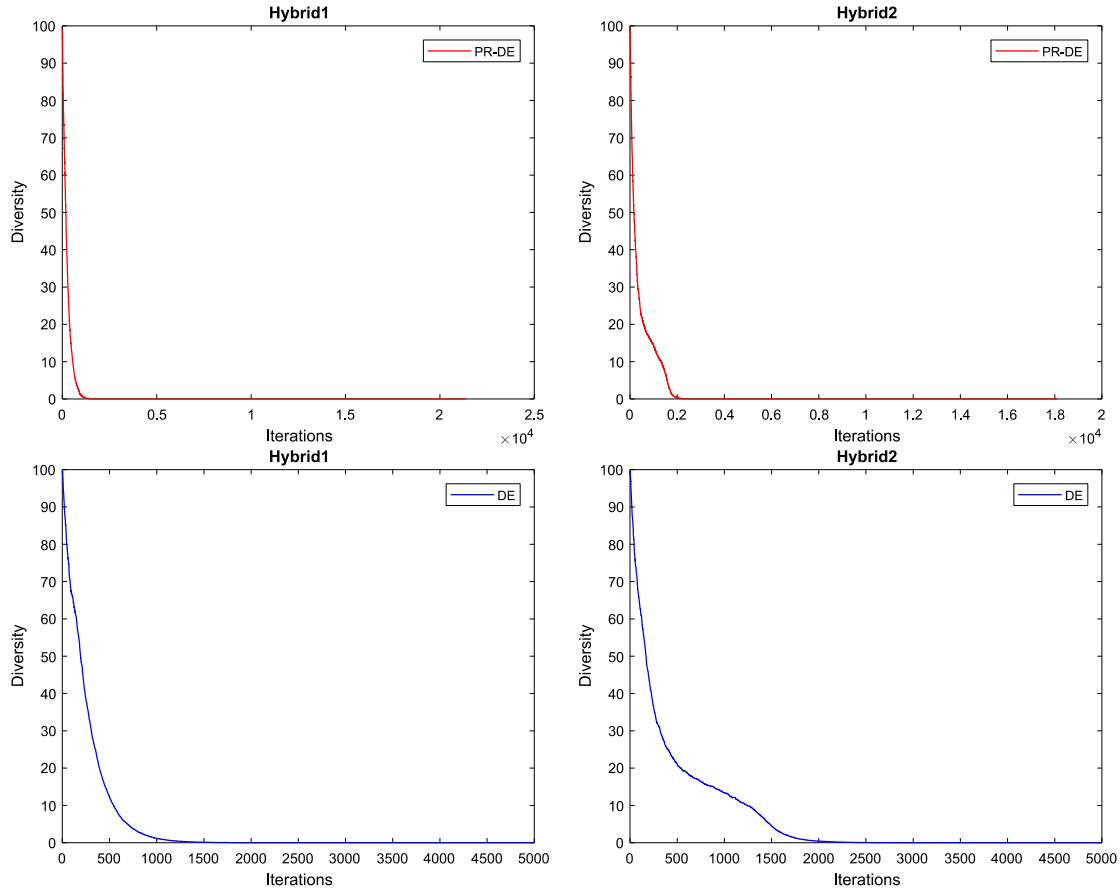

Fig. 9. Population diversity through iterations for hybrid functions f_{20} *and* f_{21}.

Table A.1
Multimodal test benchmark functions considered in the experiments.

f_i	Function	Equation	Bounds	Dim	Optimum
f_1	Ackley	$f(x) = -20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)} + 20 + e$	$[-30, 30]^n$	$n = 30$	$f(\mathbf{x}^*) = 0;$ $\mathbf{x}^* = (0, \ldots, 0)$
f_2	Dixon Price	$f(x) = (x_i - 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2$	$[-10, 10]^n$	$n = 30$	$f(\mathbf{x}^*) = 0;$ $\mathbf{x}^* = 2^{-\frac{2^i-2}{2^i}}$ for $i = 1, \ldots, n$
f_3	Infinity	$f(x) = \sum_{i=1}^{n} x_i^6 \left[\sin\left(\frac{1}{x_i}\right) + 2\right]$	$[-1, 1]^n$	$n = 30$	$f(\mathbf{x}^*) = 0;$ $\mathbf{x}^* = (0, \ldots, 0)$
f_4	Levy	$f(x) = \cos 2(\pi w_1) + \sum_{i=1}^{n-1}(w_i - 1)^2(1 + 10\sin \pi w_i + 1) + (w_n - 1)^2(1 + sin^2 2\pi w_n)$ $w_i = 1 + (\frac{x_i + 1}{4})$	$[-10, 10]^n$	$n = 30$	$f(\mathbf{x}^*) = 0;$ $\mathbf{x}^* = (1, \ldots, 1)$
f_5	Mishra11	$f(\mathbf{x}) = \left[\frac{1}{n}\sum_{i=1}^{n}\lvert x_i\rvert - \left(\prod_{i=1}^{n}\lvert x_i\rvert\right)^{\frac{1}{n}}\right]^2$	$[-10, 10]^n$	$n = 30$	$f(\mathbf{x}^*) = 0;$ $\mathbf{x}^* = (0, \ldots, 0)$
f_6	Multimodal	$f(x) = \sum_{i}^{n}\lvert x_i\rvert \times \prod_{i}^{n}\lvert x_i\rvert$	$[-10, 10]^n$	$n = 30$	$f(\mathbf{x}^*) = 0;$ $\mathbf{x}^* = (0, \ldots, 0)$

(*continued on next page*)

Appendix A

The following Tables contain detailed information on the optimization functions utilized throughout this work. The functions are separated by classification, multimodal (Table A.1), unimodal (Table A.2), hybrid (Table A.3) and shifted (Table A.4).

Appendix B

Cassini 1 design problem

The objective of the Cassini 1 spacecraft mission design problem is to determinate a possible trajectory that a spacecraft may take to travel from the Earth to Saturn, while minimizing the propellant required. The spacecraft is limited to only use thrust at planetary encounters. The considered planetary sequence is Earth

Table A.1 (*continued*).

f_i	Function	Equation	Bounds	Dim	Optimum
f_7	Perm2	$f(x) = \sum_{k=1}^{n} \left\{ \sum_{j}^{n} (j^k + \beta) \left[x_j{}^k - \frac{1}{j} \right] \right\}^2$	$[-n, n]^n$	$n = 30$	$f(\mathbf{x}^*) = 0$; $\mathbf{x}^* = (1, 1/2, \ldots, 1/n)$
f_8	Plateau	$f(x) = 30 + \sum_{i=1}^{n} \lfloor x_i \rfloor$	$[-5.12, 5.12]^n$	$n = 30$	$f(\mathbf{x}^*) = 30$; $\mathbf{x}^* = (0, \ldots, 0)$
f_9	Qing	$f(x) = \sum_{i=1}^{n} \left(x_i{}^2 - 1 \right)^2$	$[-500, 500]^n$	$n = 30$	$f(\mathbf{x}^*) = 0$; $x_i{}^* = \left(\pm\sqrt{i} \right)$ $i = 1, 2, \ldots, n$
f_{10}	Quartic	$f(x) = \sum_{i=1}^{n} \left\{ (ix_i)^4 + rand\,[0, 1) \right\}$	$[-1.28, 1.28]^n$	$n = 30$	$f(\mathbf{x}^*) = 0$; $\mathbf{x}^* = (0, \ldots, 0)$
f_{11}	Rosenbrock	$f(x) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-5, 10]^n$	$n = 30$	$f(\mathbf{x}^*) = 0$; $\mathbf{x}^* = (1, \ldots, 1)$
f_{12}	Schwefel21	$f(x) = \max \lvert x_i \rvert$	$[-100, 100]^n$	$n = 30$	$f(\mathbf{x}^*) = 0$; $\mathbf{x}^* = (0, \ldots, 0)$
f_{13}	Schwefel22	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-100, 100]^n$	$n = 30$	$f(\mathbf{x}^*) = 0$; $\mathbf{x}^* = (0, \ldots, 0)$
f_{14}	Styblinski Tang	$f(x) = \frac{1}{2} \sum_{i=1}^{n} \left(x_i^4 - 16x_i^2 + 5x_i \right)$	$[-5, 5]^n$	$n = 30$	$f(\mathbf{x}^*) = -39.1659n$; $\mathbf{x}^* = (-2.90, \ldots, 2.90)$

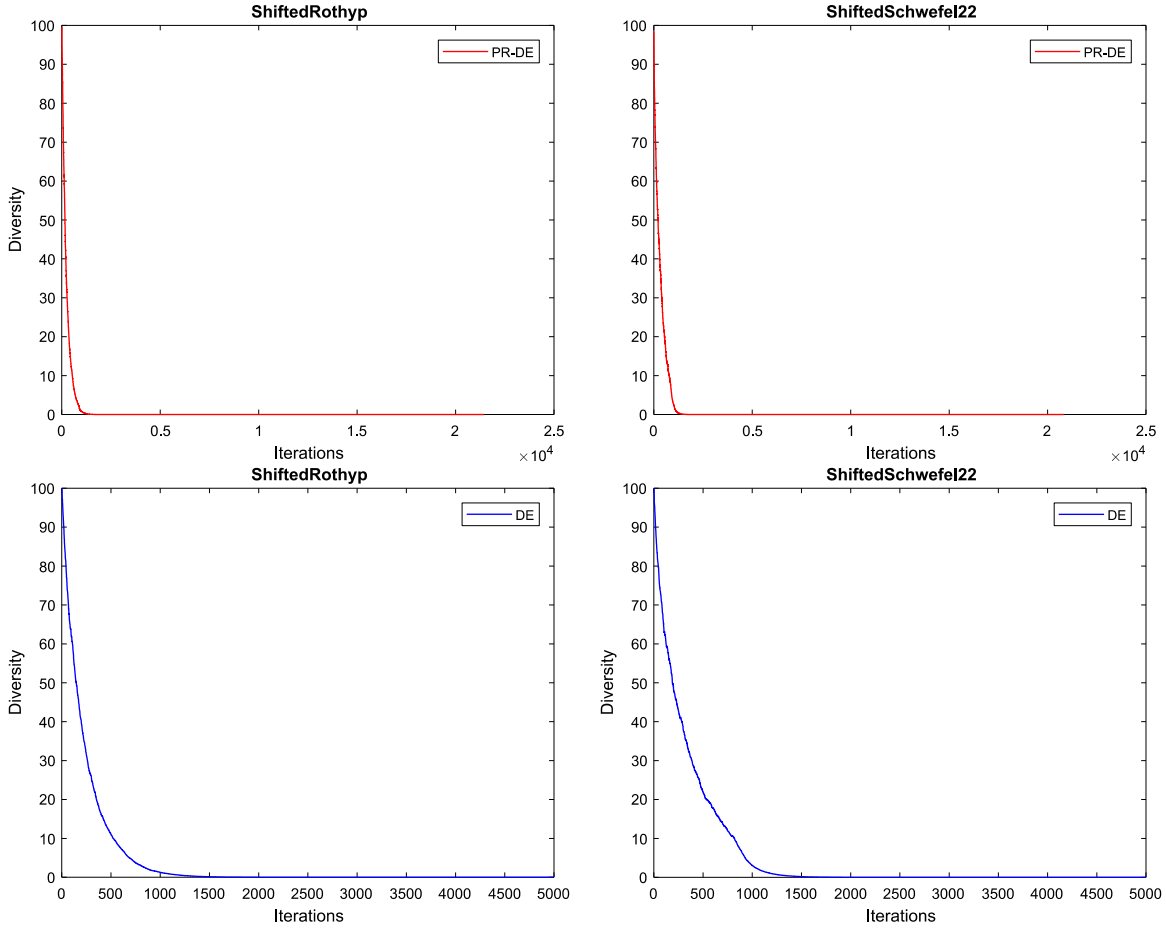

Fig. 10. Population diversity through iterations for shifted functions f_{25} *and* f_{27}.

-> Venus -> Venus -> Earth -> Jupiter -> Saturn. Important parameters of the mission are the pericenter radius of Saturn (108,950 km), and its eccentricity value (0.98).

This problem considers six design variables to optimize:

Variable	Units	Lower limits	Upper limits
T0	MJD2000	−1000	0
T1	Days	30	400
T2	Days	100	470
T3	Days	30	400
T4	Days	400	2000
T5	Days	1000	6000

Table A.2
Unimodal test benchmark functions considered in the experiments.

f_i	Function	Equation	Bounds	Dim	Optimum
f_{15}	Rothyp	$f(x)=\sum_{i=1}^{n}\sum_{j=1}^{i}x_j^2$	$[-65.536, 65.536]^n$	$n=30$	$n=30\mathbf{x}^*=(0,\ldots,0)$
f_{16}	Schwefel2	$f(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_i\right)^2$	$[-100, 100]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(0,\ldots,0)$
f_{17}	Sphere	$f(x)=\sum_{i=1}^{n}x_i^2$	$[-5, 5]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=0,\ldots,0$
f_{18}	Sum Squares	$f(x)=\sum_{i=1}^{n}ix_i^2$	$[-10, 10]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(0,\ldots,0)$
f_{19}	Sum ofDifferent Powers	$f(x)=\sum_{i=1}^{n}\lvert x_i\rvert^{i+1}$	$[-1, 1]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(0,\ldots,0)$

Table A.3
Hybrid test benchmark functions considered in the experiments.

f_i	Function	Equation	Bounds	Dim	Optimum
f_{20}	Hybrid1	$f(x)=Rastringin+Schwefel22+Sphere$	$[-100, 100]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(0,\ldots,0)$
f_{21}	Hybrid2	$f(x)=Griewank+Rastringin+Rosenbrock$	$[-100, 100]^n$	$n=30$	$f(\mathbf{x}^*)=n-1;$ $\mathbf{x}^*=(0,\ldots,0)$
f_{22}	Hybrid4	$f(x)=Ackely+Griewank+Rastringin+Rosenbrock+Schwefel22$	$[-100, 100]^n$	$n=30$	$f(\mathbf{x}^*)=n-1;$ $\mathbf{x}^*=(0,\ldots,0)$

Table A.4
Shifted test benchmark functions considered in the experiments.

f_i	Function	Equation	Bounds	Dim	Optimum
f_{23}	Shifted Ackley	$f(x)=-20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i-10)^2}}-e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi(X_i-10))}+20+e$	$[-20, 40]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(10,\ldots,10)$
f_{24}	Shifted Rosenbrock	$f(x)=\sum_{i=1}^{n-1}\left[100((x_{i+1}-100)-(x_i-100)^2)^2+((x_i-100)-1)^2\right]$	$[96, 111]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(101,\ldots,101)$
f_{25}	Shifted Rothyp	$f(x)=\sum_{i=1}^{n}\sum_{j=1}^{i}(x_j-20)^2$	$[-45.536,^n$ $85.536]$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(20,\ldots,20)$
f_{26}	Shifted Schwefel2	$f(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}(x_i-100)\right)^2$	$[0, 200]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(100,\ldots,100)$
f_{27}	Shifted Schwefel22	$f(x)=\sum_{i=1}^{n}\lvert(x_i-25)\rvert+\prod_{i=1}^{n}\lvert(x_i-25)\rvert$	$[-75, 125]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(25,\ldots,25)$
f_{28}	Shifted sphere	$f(x)=\sum_{i=1}^{n}(x_i-20)^2$	$[14.88, 25.12]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(20,\ldots,20)$
f_{29}	Shifted Sum2	$f(x)=\sum_{i=1}^{n}i(x_i-30)^2$	$[20, 40]^n$	$n=30$	$f(\mathbf{x}^*)=0;$ $\mathbf{x}^*=(30,\ldots,30)$

The constraints for the fly-by pericenters of the considered planetary sequence are:

Radius pericenter 1 > 6351.8 km

Radius pericenter 2 > 6351.8 km

Radius pericenter 3 > 6778.1 km

Radius pericenter 4 > 671,492 km

SAGAS design problem

The objective of the SAGAS spacecraft mission design problem is to do a deltaV-EGA maneuver to reach a fly-by with Jupiter in 50 astronomical units (AU). Unlike the Cassini 1 design problem, the spacecraft thrusts are not limited and can be used at any point of each trajectory section. The objective function considered is the minimization of the overall mission length.

This problem considers 12 design variables to optimize:

Variable	Units	Lower limits	Upper limits
T0	MJD2000	7000	9100
VINF	km/s	0	7
U	n/a	0	1
V	n/a	0	1
T1	days	50	2000
T2	days	300	2000
ETA1	n/a	0.01	0.9
ETA2	n/a	0.01	0.9
RP1	n/a	1.05	7
RP2	n/a	8	500
BINCL1	rad	-Pi	Pi
BINCL2	rad	-Pi	Pi

The constraints deal with the limited fuel capacity, and the launcher performance:

deltaV1 + deltaV2 < 1.782

deltaV = deltaV1 + delta V2 + delta VInfinity < 6.782 km/s

References

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

[2] J. Xu, J. Zhang, Exploration-exploitation tradeoffs in metaheuristics: Survey and analysis, in: Proc. 33rd Chinese Control Conf. CCC 2014, 2014, pp. 8633–8638, http://dx.doi.org/10.1109/ChiCC.2014.6896450.

[3] M. Črepiňsek, S.-H. Liu, M. Mernik, Exploration and exploitation in evolutionary algorithms: A survey, ACM Comput. Surv. Artic. 45 (33) (2013) 1–33, http://dx.doi.org/10.1145/2480741.2480752.

[4] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: A gravitational search algorithm, Inf. Sci. (Ny). 179 (13) (2009) 2232–2248, http://dx.doi.org/10.1016/j.ins.2009.03.004.

[5] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Sci. (80-.) 220 (4598) (1983) 671–680, http://dx.doi.org/10.1126/science.220.4598.671.

[6] E. Cuevas, A. Echavarría, M.A. Ramírez-Ortegón, An optimization algorithm inspired by the states of matter that improves the balance between exploration and exploitation, Appl. Intell. 40 (2) (2014) 256–272, http://dx.doi.org/10.1007/s10489-013-0458-0.

[7] K.S. Tang, K.F. Man, S. Kwong, Q. He, Genetic algorithms and their applications, IEEE Signal Process. Mag. 13 (6) (1996) 22–37, http://dx.doi.org/10.1109/79.543973.

[8] R. Storn, K. Price, Differential evolution – A simple and efficient Heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359, http://dx.doi.org/10.1023/A:1008202821328.

[9] Jingqiao Zhang, A.C. Sanderson, JADE: Self-adaptive differential evolution with fast and reliable convergence performance, in: 2007 IEEE Congress on Evolutionary Computation, 2007, pp. 2251–2258, http://dx.doi.org/10.1109/CEC.2007.4424751.

[10] H.-G. Beyer, H.-G. Beyer, H.-P. Schwefel, H.-P. Schwefel, Evolution strategies – A comprehensive introduction, Nat. Comput. 1 (1) (2002) 3–52, http://dx.doi.org/10.1023/A:1015059928466.

[11] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95 - International Conference on Neural Networks, Vol. 4, 1995, pp. 1942–1948, http://dx.doi.org/10.1109/ICNN.1995.488968.

[12] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Global Optim. 39 (3) (2007) 459–471, http://dx.doi.org/10.1007/s10898-007-9149-x.

[13] X.-S. Yang, Firefly algorithms for multimodal optimization, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 5792, LNCS, 2009, pp. 169–178.

[14] A.H. Gandomi, X.-S. Yang, A.H. Alavi, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, Eng. Comput. 29 (1) (2013) 17–35, http://dx.doi.org/10.1007/s00366-011-0241-y.

[15] E. Cuevas, M. Cienfuegos, D. Zaldívar, M. Pérez-Cisneros, A swarm optimization algorithm inspired in the behavior of the social-spider, Expert Syst. Appl. 40 (16) (2013) 6374–6384, http://dx.doi.org/10.1016/j.eswa.2013.05.041.

[16] A. Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm, Comput. Struct. 169 (2016) 1–12, http://dx.doi.org/10.1016/j.compstruc.2016.03.001.

[17] S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowledge-Based Syst. 89 (2015) 228–249, http://dx.doi.org/10.1016/j.knosys.2015.07.006.

[18] X.-S. Yang, A new metaheuristic bat-inspired algorithm, in: Studies in Computational Intelligence, Vol. 284, 2010, pp. 65–74.

[19] K. Sörensen, Metaheuristics-the metaphor exposed, Int. Trans. Oper. Res. 22 (1) (2015) 3–18, http://dx.doi.org/10.1111/itor.12001.

[20] B. Morales-Castañeda, D. Zaldívar, E. Cuevas, F. Fausto, A. Rodríguez, A better balance in metaheuristic algorithms: Does it exist? Swarm Evol. Comput. 54 (2020) 100671, http://dx.doi.org/10.1016/j.swevo.2020.100671.

[21] M.N.M. Salleh others, Exploration and Exploitation Measurement in Swarm-Based Metaheuristic Algorithms: An Empirical Analysis, 1, (2018) 24–32.

[22] H.-P. Kriegel, E. Schubert, A. Zimek, The (black) art of runtime evaluation: Are we comparing algorithms or implementations? Knowl. Inf. Syst. 52 (2) (2017) 341–378, http://dx.doi.org/10.1007/s10115-016-1004-2.

[23] S. Cheng, Y. Shi, Q. Qin, Q. Zhang, R. Bai, Population diversity maintenance in brain storm optimization algorithm, J. Artif. Intell. Soft Comput. Res. 4 (2) (2014) 83–97, http://dx.doi.org/10.1515/jaiscr-2015-0001.

[24] X.-S. Yang, S. Deb, S. Fong, Metaheuristic algorithms: Optimal balance of intensification and diversification, Appl. Math. Inf. Sci. 8 (3) (2014) 977–983, http://dx.doi.org/10.12785/amis/080306.

[25] C. Huang, Y. Li, X. Yao, A survey of automatic parameter tuning methods for metaheuristics, IEEE Trans. Evol. Comput. 24 (2) (2020) 201–216, http://dx.doi.org/10.1109/TEVC.2019.2921598.

[26] D. Johnson, A theoretician's guide to the experimental analysis of algorithms, in: Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges, 2002, pp. 215–250.

[27] N. Hansen, A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, Evol. Comput. 9 (2) (2001) 159–195, http://dx.doi.org/10.1162/106365601750190398.

[28] R. Tanabe, A. Fukunaga, Success-history based parameter adaptation for differential evolution, 2013 IEEE Congr. Evol. Comput. CEC 2013 (3) (2013) 71–78, http://dx.doi.org/10.1109/CEC.2013.6557555.

[29] R. Tanabe, A.S. Fukunaga, Improving the search performance of SHADE using linear population size reduction, in: 2014 IEEE Congress on Evolutionary Computation (CEC), 2014, pp. 1658–1665, http://dx.doi.org/10.1109/CEC.2014.6900380.

[30] L. Cui, G. Li, Q. Lin, J. Chen, N. Lu, Adaptive differential evolution algorithm with novel mutation strategies in multiple sub-populations, Comput. Oper. Res. 67 (2016) 155–173, http://dx.doi.org/10.1016/j.cor.2015.09.006.

[31] F.G. Lobo, C.F. Lima, Z. Michalewicz, Parameter Setting in Evolutionary Algorithms, Vol. 54, (54) Heidelberg: Springer Berlin Heidelberg, Berlin, 2007.

[32] E.S. Skakov, V.N. Malysh, Parameter meta-optimization of metaheuristics of solving specific NP-hard facility location problem, J. Phys. Conf. Ser. 973 (1) (2018) http://dx.doi.org/10.1088/1742-6596/973/1/012063.

[33] A.E. Eiben, J.E. Smith, Introduction To Evolutionary Computing, Vol. 33, (5/6) Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[34] G. Karafotias, M. Hoogendoorn, A.E. Eiben, Parameter control in evolutionary algorithms: Trends and challenges, IEEE Trans. Evol. Comput. 19 (2) (2015) 167–187, http://dx.doi.org/10.1109/TEVC.2014.2308294.

[35] M. Birattari, Tuning Metaheuristics, Vol. 197, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[36] V. Nannen, A.E. Eiben, A method for parameter calibration and relevance estimation in evolutionary algorithms, in: GECCO 2006 - Genet. Evol. Comput. Conf. Vol. 1, 2006, pp. 183–190.

[37] F. Hutter, H.H. Hoos, K. Leyton-Brown, T. Stützle, Paramils: An automatic algorithm configuration framework, J. Artificial Intelligence Res. 36 (2009) 267–306, http://dx.doi.org/10.1613/jair.2808.

[38] F. Hutter, H.H. Hoos, K. Leyton-Brown, K.P. Murphy, An experimental investigation of model-based parameter optimisation: SPO and beyond, Proc. 11th Annu. Genet. Evol. Comput. Conf. GECCO-2009, 2009, pp. 271–278, doi=http://dx.doi.org/10.1145/1569901.1569940.

[39] F. Hutter, H.H. Hoos, K. Leyton-Brown, Sequential model-based optimization for general algorithm configuration, in: International Conference on Learning and Intelligent Optimization, 2011, pp. 507–523.

[40] P.A. Consoli, Y. Mei, L.L. Minku, X. Yao, Dynamic selection of evolutionary operators based on online learning and fitness landscape analysis, Soft Comput. 20 (10) (2016) 3889–3914, http://dx.doi.org/10.1007/s00500-016-2126-x.

[41] F. Fausto, A. Reyna-Orta, E. Cuevas, Á.G. Andrade, M. Perez-Cisneros, From ants to whales: metaheuristics for all tastes, Artif. Intell. Rev. 53 (1) (2020) 753–810, http://dx.doi.org/10.1007/s10462-018-09676-2.

[42] G. Li others, A novel hybrid differential evolution algorithm with modified CoDE and JADE, Appl. Soft Comput. J. 47 (2016) 577–599, http://dx.doi.org/10.1016/j.asoc.2016.06.011.

[43] Y. Wang, Z. Cai, Q. Zhang, Differential evolution with composite trial vector generation strategies and control parameters, IEEE Trans. Evol. Comput. 15 (1) (2011) 55–66, http://dx.doi.org/10.1109/TEVC.2010.2087271.

[44] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417, http://dx.doi.org/10.1109/TEVC.2008.927706.

[45] Y. Cai, J. Wang, Differential evolution with neighborhood and direction information for numerical optimization, IEEE Trans. Cybern. 43 (6) (2013) 2202–2215, http://dx.doi.org/10.1109/TCYB.2013.2245501.

[46] F. Fagan, J.H. Van Vuuren, A unification of the prevalent views on exploitation, exploration, intensification and diversification, Int. J. Metaheuristics 2 (3) (2013) 294, http://dx.doi.org/10.1504/IJMHEUR.2013.056407.

[47] M.Z. Ali, N.H. Awad, P.N. Suganthan, R.G. Reynolds, An adaptive multipopulation differential evolution with dynamic population reduction, IEEE Trans. Cybern. 47 (9) (2017) 2768–2779, http://dx.doi.org/10.1109/TCYB.2016.2617301.

[48] B. M'Hamdi, M. Teguar, A. Mekhaldi, Optimal design of corona ring on HV composite insulator using PSO approach with dynamic population size, IEEE Trans. Dielectr. Electr. Insul. 23 (2) (2016) 1048–1057, http://dx.doi.org/10.1109/TDEI.2015.005383.

[49] J. Aalto, J. Lampinen, A population adaptation mechanism for differential evolution algorithm, Proc. - 2015 IEEE Symp. Ser. Comput. Intell. SSCI 2015, 2015, pp. 1514–1521, http://dx.doi.org/10.1109/SSCI.2015.214.

[50] S. Limmer, D. Fey, Investigation of strategies for an increasing population size in multi-objective CMA-ES, 2016 IEEE Congr. Evol. Comput. CEC 2016 (2016) 476–483, http://dx.doi.org/10.1109/CEC.2016.7743832.

[51] P. Bujok, J. Tvrdik, Enhanced individual-dependent differential evolution with population size adaptation, in: 2017 IEEE Congr. Evol. Comput. CEC 2017 - Proc, 2017, pp. 1358–1365, http://dx.doi.org/10.1109/CEC.2017.7969462.

[52] M. Gomespereira De Lacerda, H. Deandrade Amorim Neto, T. Bernardaludermir, H. Kuchen, F. Buarquede Lima Neto, Population size control for efficiency and efficacy optimization in population based metaheuristics, in: 2018 IEEE Congr. Evol. Comput. CEC 2018 - Proc, 2018, http://dx.doi.org/10.1109/CEC.2018.8477792.

[53] F. Wilcoxon, Individual comparisons of grouped data by ranking methods, J. Econ. Entomol. 39 (2) (1946) 269–270, http://dx.doi.org/10.1093/jee/39.2.269.
[54] S. García, D. Molina, M. Lozano, F. Herrera, A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the cec'2005 special session on real parameter optimization, J. Heuristics 15 (6) (2009) 617–644.
[55] N. Hansen, The CMA evolution strategy: A tutorial, 2016, pp. 1–39, [Online]. Available: http://arxiv.org/abs/1604.00772.
[56] S. Das, P.N. Suganthan, Differential evolution: A survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (1) (2011) 4–31, http://dx.doi.org/10.1109/TEVC.2010.2059031.
[57] F. Marini, B. Walczak, Particle swarm optimization (PSO), A tutorial, Chemom. Intell. Lab. Syst. 149 (2015) 153–165, http://dx.doi.org/10.1016/j.chemolab.2015.08.020.
[58] K. Hussain, M.N. Mohd Salleh, S. Cheng, R. Naseem, Common benchmark functions for metaheuristic evaluation: A review, JOIV Int. J. Informatics Vis. 1 (4–2) (2017) 218, http://dx.doi.org/10.30630/joiv.1.4-2.65.
[59] D. Izzo, Parameter Setting in Evolutionary Algorithms, Vol. 54, (9) Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.
[60] B. Addis, A. Cassioli, M. Locatelli, F. Schoen, A global optimization method for the design of space trajectories, Comput. Optim. Appl. 48 (3) (2011) 635–652, http://dx.doi.org/10.1007/s10589-009-9261-6.
[61] G. Stracquadanio, A. La Ferla, M. De Felice, G. Nicosia, Design of robust space trajectories, in: M. Bramer, M. Petridis, L. Nolle (Eds.), Research and Development in Intelligent Systems XXVIII, Springer London, London, 2011, pp. 341–354.