Are evolutionary algorithm competitions characterizing landscapes appropriately?

P. Caamaño
Grupo Integrado de Ingeniería
Escuela Politécnica Superior
Universidade da Coruña,
Spain
pcsobrino@udc.es

J.A. Becerra
Grupo Integrado de Ingeniería
Escuela Politécnica Superior
Universidade da Coruña,
Spain
ronin@udc.es

F. Bellas
Grupo Integrado de Ingeniería
Escuela Politécnica Superior
Universidade da Coruña,
Spain
fran@udc.es

R.J. Duro
Grupo Integrado de Ingeniería
Escuela Politécnica Superior
Universidade da Coruña,
Spain
richard@udc.es

ABSTRACT

Nowadays, there exist some very well-known benchmark function sets used by researchers to test the behavior of an evolutionary algorithm and to compare it to others. These functions are classified using characteristics like dimensionality, modality or separability but, typically, there is a lack of an in depth analysis of them. Without such analysis, many times, it is not clear why an evolutionary algorithm performs better over some functions and worse over others and, therefore, researchers focus just on how many functions are solved and how fast they are. This is a problem for an evolutionary algorithm user, who usually choose some state of the art algorithm without being sure about the correctness of their decision. This paper goes in the direction of establishing methods for a more in depth analysis of fitness landscapes of benchmark functions and showing their need, so more conclusions can be obtained about what scenarios are adequate or inappropriate for an evolutionary algorithm. In particular, some new features are presented to study separability and modality in more detail than generally known and it will be shown, using three different evolutionary algorithms, the need of them to understand why things happen the way they do.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Searchs—*Heuristic methods*

General Terms

Algorithms

Keywords

Evolutionary computation, fitness landscapes, modality analysis, separability analysis, differential evolution

1. INTRODUCTION

Currently, researchers in the field of Evolutionary Algorithms (EAs) are very interested in competitions where new implementations are evaluated. These competitions take place during EA workshops and conferences, such as GECCO (EvoDOP-2007) or CEC [18, 10, 20], and researchers from different disciplines use them as a reference because of their interest in the application of this type of algorithms to solve their problems. Usually, these EA users perform their algorithm selections by following the results published in these competitions, which are typically focused on average performance measures. Once they have selected an EA, the common procedure they follow is to tune its configuration and apply it to try to solve their problem until a successful solution is obtained [23, 13]. If its initial approach fails, the next step is to attempt to update it with a newer EA after consulting the literature of the topic, like authors do in [19], probably selecting an algorithm that obtains the best results on average and which is considered the winner in some other competition. This selection could prove to be completely wrong if the algorithm fails in a particular feature required for solving the specific problem the user faces. This is a highly time-consuming and frustrating trial and error process.

In order to try to minimize this problem there is no doubt that it is necessary to increase our knowledge about how an EA behaves. One way of doing this is increasing the detail of the analysis of the benchmark functions used to observe the behavior of EAs. Although, the comparisons of the algorithms are performed fairly by specifying a common stopping criterion, problem size, initialization scheme, etc. and even though these sets are very complete, we have detected shortcomings in the way they are characterized. On those benchmark sets the functions are usually classified into binary classes according to their separability and modality. The success and the performance of an EA over a function are closely related with these two features, on the manner on which its search strategy is adapted to the morphology of the fitness landscape of a function. So, a finer grained analysis of the landscapes features will allow the designer to obtain more reliable and usable conclusions about the behavior of their algorithms and thus the users will be able to choose the right algorithms for their applications more effectively. But as we will show later in this work, the binary classification used on the competitions could produce misleading conclusions about the performance of the EAs. Thus, a deeper analysis the features that are relevant to describe EA behavior is necessary, as the one provided by the competitions is very general [6].

This papers goes in this direction proposing additional features of separability and modality to be analyzed. To determine and test the features that are proposed here, we have chosen a benchmark function set [5] with functions that are commonly used in EA competitions classified according their separability and modality. The benchmark set contains 36 scalable and non-scalable functions that allow us to study the differences between the fitness landscapes in depth so as to determine what makes them different and link these differences to EA performance.

The remainder of the paper is structured as follows, Section 2 explain the experimental procedure followed in this work, Section 3 explains and describes the set of features that we propose and it will be considered to analyze the behavior of the algorithms. In this section several examples illustrate the importance of the deeper analysis relating the behavior and configuration of the algorithms with the features of the fitness landscape of the functions. Finally some conclusions are presented in Section 4.

2. EXPERIMENTAL PROCEDURE

The main objective of this work is to analyze features like separability and modality and how they help to explain how an EA behaves depending on the presence of them on a function fitness landscape. As we have said in the previous section, our work had shown that the typical binary classification is not enough to obtain reliable conclusions about that. In order to show that the analysis of the fitness landscape is very useful for the study of EA behavior, three EAs were chosen. Two of them are the winners of most of the competitions: Differential Evolution (DE) [17] and the Covariance Matrix Adaptation - Evolutionary Strategy (CMA-ES) [3]. Additionally a Real-Coded Genetic Algorithm (RCGA) [9] is chosen as an EA reference. To analyze the behavior of these algorithms, their parameters are set as their authors recommend. For each algorithm-function-dimension combination, 25 independent runs were executed. The scalable function set was analyzed considering 10, 30 and 50 dimensions as is typical of EA competitions. The stop criteria of the runs is based on the maximum number of function evaluations (FEs) and it is set to $10000 \cdot n$, where n is the dimension of the problem. To study the results provided by the algorithms we have proposed the *Combined Error and Performance Measure* (CPEM). It provides combined results on the error level and the performance of the algorithms at a glance. It is calculated as follows:

$$CPEM = \begin{cases} \varepsilon \cdot \dfrac{FEs}{FEs_{max}} & \text{if } FEs \leq FEs_{max} \\ \text{absolute error} & \text{otherwise} \end{cases} \quad (1)$$

Where, ε is the threshold to consider that a function is solved, in this work it has set to 10^{-6}.

All the results on tables 3 and 4 are provided in terms of average CPEM over the 25 independent runs. In these table the lowest CPEM value corresponds with the better algorithm for each function.

3. FITNESS LANDSCAPE CHARACTERIZATION

In this section the classification given in the benchmark sets of the competitions [18, 10, 21, 20] are analyzed. Some solutions for the detected problems are proposed.

3.1 Separability

Separability refers to the dependencies presented between the parameters of a function and it is related with the biological concept of epistasis [16, 14]. According to separability, the competitions' benchmark sets are usually divided into Separable functions and Non-Separable. Separable functions [12] are those where all the parameters are independent and, as a consequence, the optimization process of an n-dimensional Separable function ($f(\mathbf{x}) : \mathbb{R}^n \longrightarrow \mathbb{R}$) could be divided into n 1-D optimization process over each parameter x_i, where $0 \leq i < n$. Non-Separable functions [3] are those where dependencies are present and all the parameters should be optimized during the same process beacuse of the relationships between them.

Two facts lead us to analyze deeper the separability of the functions. On one hand, as a first step the three selected algorithms were analyzed using the typical binary classification. Figure 2(a) shows the results provided by the RCGA implementation of this work. As shown, it is difficult to discriminate what is happening with the behavior of the RCGA in the Non-Separable function set, where functions solved are mixed with functions not solved. On the other hand, sometimes it is difficult to classify a function because there is no consensus in the bibliography. For example, Ackley's function [1] sometimes is classified as Separable [22] and another times as Non-Separable [2].

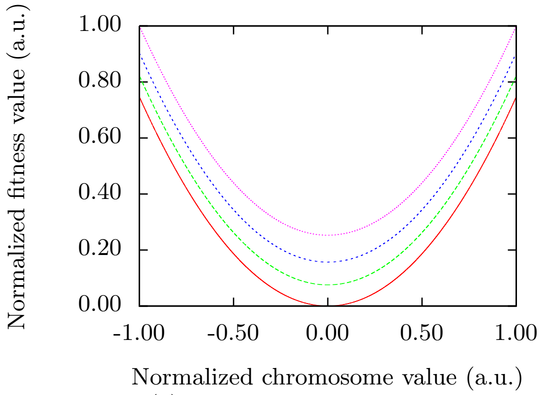
After analyzing the results of the RCGA over *Non-Separable* functions in detail, we have realized that there are different types of functions within this subset. It includes functions that are separable, but not linearly, and other functions that are Non-separable. So, we propose a different classification in terms of separability that classifies functions into linearly separable functions (L-Separable), non-linearly separable functions (NL-Separable) [24] and Non-Separable functions. On table 3 and table 4 after the name of the functions a L, NL or N refers to L-Separable, NL-Separable and Non-Separable functions, respectively.

To characterize an objective function in terms of these categories is not easy, and an empirical method based on the graphical representation of the function have been developed. The basic idea is to fix the values of *n-1* parameters of an n parameter function and iterate the remaining parameter between the low and the high bounds for different values of the fixed parameters. Using the 2-D plots of these runs and their comparative behavior, a decision on the type of function can be made. Thus, a function is linearly separable (*L-separable*) if, for different values of these *n-1* parameters, it preserves the same graph shape shifted in the y-axis (see Fig. 1(a)). Following this reasoning, non-linearly separable functions (*NL-separable* hereafter) are those functions with different shapes (different search space) but with the optimum always in the same point (see Fig. 1(b)); and *non-separable* functions are those with different shape and different optimum (see Fig. 1(c)). This method allow us to classify the functions into three different categories. It must be highlighted the case of the Griewank function. Its separability changes as we increase the dimensions of the problem, being Non-Separable when we deal with 10 dimensions and NL-Separable in the other two cases. This is due the definition of the function, as the authors explain in [11] the function becomes smoother as the dimensions increases and it also affects to its separability.

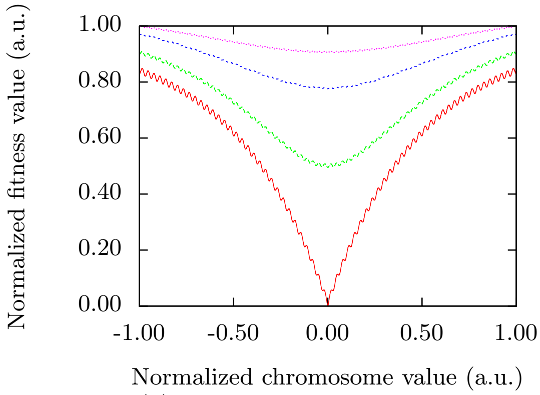
The differences between the functions are not only from the mathematic and analytical point of view. The behavior and the success of an EA is closely related with the type of separability. The process to optimize a L-Separable function could performs over one parameter at each time. Although, as the authors said in [24] a NL-Separable function the parameters presents non-linear dependencies between them, the optimal value of each parameter could be obtained over an independent optimization process independently of all other parameters. This does not happen on Non-Separable functions, where the goodness of a parameters depends on the value of some other parameters of the function.

Figure 2(b) displays the behavior of the RCGA again, but now in terms of this classification and its behavior can now be clearly discriminated. This algorithm performs successfully in those functions that are separable, either linearly and non-linearly, and its behavior degrades on Non-Separable functions. These results are due the fact that this RCGA implementation does not take into account the dependencies between parameters during the optimization process. The use of this dependencies is highly recommended over Non-Separable functions as the CMA-ES [3] performs. This algorithm approximates a covariance matrix of the function parameters representing the relationships between them. This successful behavior could be shown in the results tables (Table 3 and Table 4) and in the Figure 3 where the results provided by the three algorithms over the Non-Separable function set are shown. In this figure, it could be shown that the results of the DE are very similar (in terms of successful runs) than the CMA-ES results. Although the DE does not provided a mechanism to use the dependencies between variables to generate the new individuals during the evolutionary process, the behavior of this algorithm, as occurs in every EA, is governed by several parameters. One of them, the cross-over rate (CR), allows to determine how many parameters changes at each generation. Low values of CR causes few parameters changes, this is a benefit for L and NL-Separable functions. The contrary occurs when the CR is set to a high value, being a benefit for Non-Separable functions. In this work the CR parameter is set to 0.1 to L and NL-Separable functions and to 0.9 to Non-Separable function, as the authors recommend in [15].

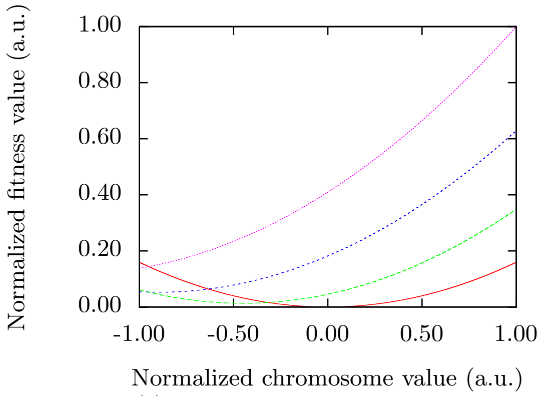
To sum up, separability is a feature highly used when analyzing the behavior of EAs. It is closely related with the search process carried out by the evolutionary operators. As it was shown in this section, the classification made in previous works is sometimes confused and mix functions, such us NL-Separable and Non-Separable, into the same group when the behavior required to solve them is very different. The method developed for this work allow us to analyze in



(a) L-separable function



(b) NL-separable function



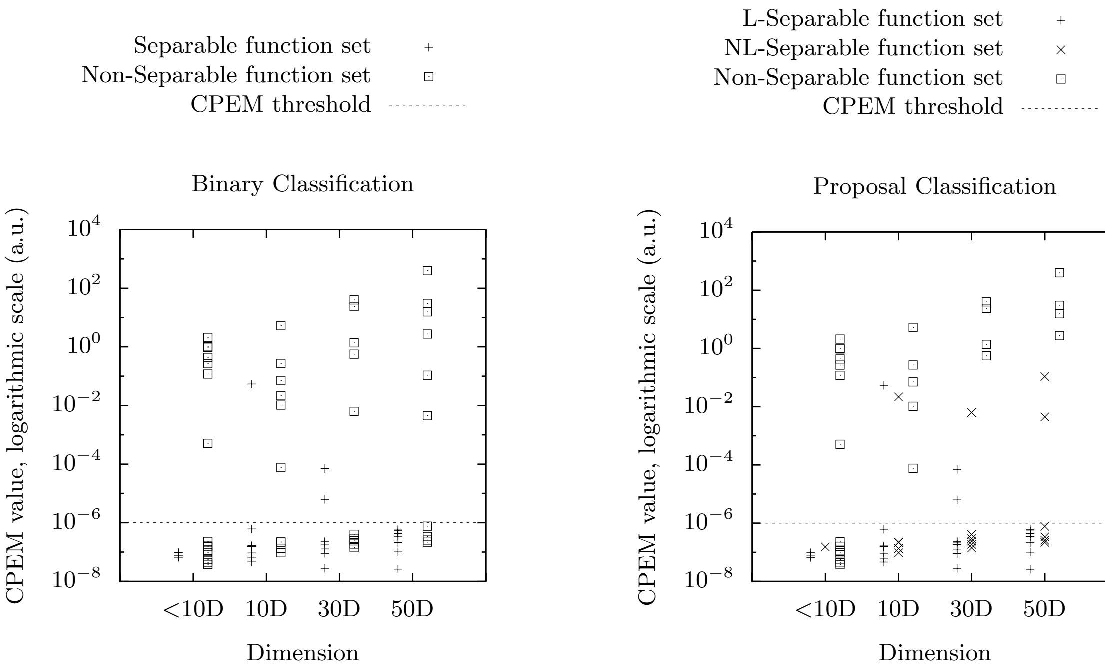
(c) Non-separable function

Figure 1: 2-D representation of the three types of separability considered

a finer grain the separability of the functions and with this information choose the best algorithm or tune up the one we have chose in the proper way.

3.2 Modality

The other most commonly used feature in EA competitions is modality. Functions are divided into unimodal and multimodal. A function is called *unimodal* if it displays a single global optimum and no local optima. On the contrary, *multimodal* functions presents several local and / or global



(a) Binary separability classification

(b) New proposal for separability classification

Figure 2: Two different proposals for separability classification.

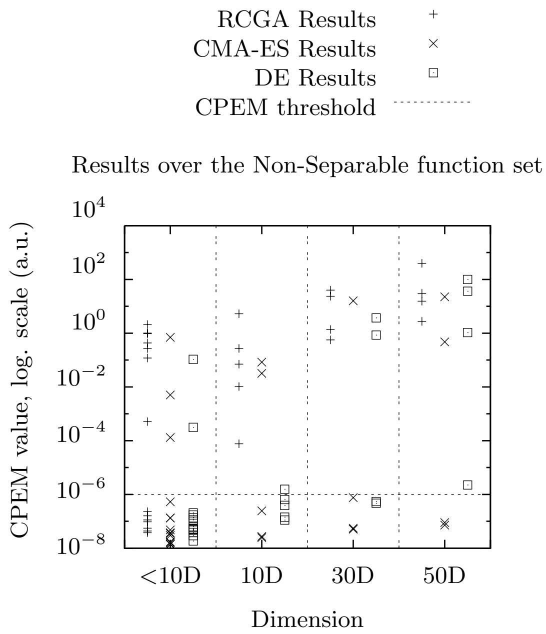


Figure 3: Results provides by the three algorithms over the Non-Separable function set.

optimum. Unimodal functions are, in general, easy to solve by a population based algorithm, and thus are mainly used to test the convergence speed of the algorithm.

Although it may seem that every unimodal function presents the same difficulty to every optimization algorithm, as Fig. 4 shows the behavior of the RCGA and the DE is different over the unimodal functions subset used in this work. There is no way to differentiate the degree of difficult of this type of function in the literature. In spite of the authors known that a needle-in-a-haystack (NiAH) [8] or a long path problem represent a challenge to EAs. To account for differences between unimodal landscapes, it was necessary to establish what landscape feature influenced the behavior of the algorithms. It turned out to be the longest path to the optimum, which is a property that is easy to measure for the different functions. To measure it a local search method was used, the length of the path from the corners of the landscape to the optimum value were measured in mean number of steps after 25 runs starting at each corner $(x_i = (x_i^1, \ldots, x_i^n), x_i^j \in -1.0, 1.0)$. The longest length is chosen as a difficulty indicator, the results of its application are shown on table 1.

In Fig. 4 the results of the DE and the RCGA over the unimodal functions are compared. As it shows, the performance of the RCGA gets worse as the path length increases. The DE implementation used here is better than the RCGA one due to its capability of dynamically adjusting the mutation step size during the optimization process. The CMA-ES implementation analyzed here also presents this capability, as the strategy to adapt the mutation step size depends on previous steps of the evolution. If during the last generations the mutations step performs on the same direction it is increased to cover the same distance in less steps, and therefore the convergence speed also increased. This strategy permits to this algorithm of being the fastest one, as it could be seen in Table 3 where the CMA-ES outperforms the results of the other two algorithms. There are three exception: *Easom*, *Perm* and *Schwefel 2.21* function. These three functions share the feature of present wide plateaus over the landscape with very few information, without this information is very difficult to the CMA-ES to generate the covariance matrix so the performance of the algorithm gets worse.

Multimodal functions are those that contain many local and global optima. Therefore, the difficulty of these functions is usually measured by their number of local optima. But, as can be seen in Fig. 5(a), it is hard to explain the behavior of the CMA-ES algorithm using this feature to characterize function difficulty. Thus, we propose other types of measures based on the theory of attraction basins proposed by the authors in [7]. As the authors said in their work, a multimodal landscape could be partitioned into N subspaces each of them corresponding to an attraction basin. An attraction basin is related to a local optima and, by definition, is a subset of points of the search space such that a local search starting at each of these points ends at the local optima. The size and the distance between the attraction basins is an estimation of their distribution and the multimodality of a n-dimensional landscape, it is very useful when $n > 2$ and a graphical representation it is not possible. An algorithm presented by the authors here [4] were applied to estimate the attraction basins distribution of the multimodal functions subset. The results obtained and used in this work to analyze the behavior of the three algorithms it is shown in table 2. In this table, the size of attraction basin of a local optima is measured as the frequency with which a random point of the search space achieve a local optima after apply over it a local search algorithm. The maximum distance is measured from every local optima to the optimum found by the attraction basins algorithm. Due to the computational complexity of this method, these results are an estimation obtained by analyzing 5-D fitness landscapes of these functions.

Different EAs are affected by different features of the distribution of the attraction basins of a fitness landscape. For example, fig. 5(b) exemplifies the fact that the performance of the CMA-ES algorithm is closely related to the maximum distance between the optimum attraction basin and all the attraction basins of the function. This behavior is due to the highly exploitative strategy of the CMA-ES. In functions where the attraction basins are spread over the fitness landscape the algorithm should explore it in order to achieve the optima attraction basin and, after reach it, exploit it to reach the optimum. The CMA-ES does not explore the search space as well as the DE, as the results shows

Table 1: Mean path length to the optimum calculated over the unimodal function subset.

Function	Path length
Perm	1060.46
Schwefel 2.21	573.48
Colville	508.30
Schwefel 1.2	453.02
Schwefel 2.22	265.15
Axis Parallel	224.97
Zakharov	220.41
Sphere Model	204.12
Matyas	188.31
SumOf	142.14
Step	126.60
Easom	24.40

Table 2: Main features of the multimodal function set.

Function	**Attr. Basin Size**		**Max.**
	Largest	**Optima**	**Distance**
Aluffi-Pentini's	$8.60{\cdot}10^{-1}$	$8.60{\cdot}10^{-1}$	$7.00{\cdot}10^{-2}$
Becker and Lago	$1.00{\cdot}10^{0}$	$1.00{\cdot}10^{0}$	$0.00{\cdot}10^{0}$
Bohachevsky 1	$7.54{\cdot}10^{-1}$	$7.54{\cdot}10^{-1}$	$9.00{\cdot}10^{-3}$
Cosine Mixture	$2.60{\cdot}10^{-2}$	$2.60{\cdot}10^{-2}$	$2.15{\cdot}10^{-1}$
Rastrigin	$8.00{\cdot}10^{-3}$	$8.00{\cdot}10^{-3}$	$4.86{\cdot}10^{-1}$
Schwefel	$2.36{\cdot}10^{-1}$	$2.96{\cdot}10^{-5}$	$9.20{\cdot}10^{-1}$
Ackley's	$1.00{\cdot}10^{-3}$	$1.00{\cdot}10^{-3}$	$5.00{\cdot}10^{-1}$
Griewank	$1.00{\cdot}10^{-3}$	$2.00{\cdot}10^{-4}$	$4.55{\cdot}10^{-1}$
Levy	$5.77{\cdot}10^{-1}$	$5.77{\cdot}10^{-1}$	$2.61{\cdot}10^{-1}$
Penalized 1	$1.28{\cdot}10^{-1}$	$1.28{\cdot}10^{-1}$	$1.13{\cdot}10^{-1}$
Penalized 2	$1.74{\cdot}10^{-1}$	$1.74{\cdot}10^{-1}$	$5.00{\cdot}10^{-2}$
Beale	$4.69{\cdot}10^{-1}$	$4.69{\cdot}10^{-1}$	$5.91{\cdot}10^{-1}$
Bohachevsky 2	$5.35{\cdot}10^{-1}$	$5.35{\cdot}10^{-1}$	$9.00{\cdot}10^{-3}$
Dekkers & Aarts	$8.79{\cdot}10^{-1}$	$8.79{\cdot}10^{-1}$	$2.91{\cdot}10^{-1}$
Goldstein Price	$5.03{\cdot}10^{-1}$	$5.03{\cdot}10^{-1}$	$4.24{\cdot}10^{-1}$
Hartman 3	$6.22{\cdot}10^{-1}$	$6.22{\cdot}10^{-1}$	$4.66{\cdot}10^{-1}$
Hartman 6	$6.25{\cdot}10^{-1}$	$2.74{\cdot}10^{-1}$	$4.50{\cdot}10^{-1}$
Kowalik	$1.28{\cdot}10^{-1}$	$2.87{\cdot}10^{-4}$	$4.27{\cdot}10^{-1}$
Rosenbrock	$3.76{\cdot}10^{-1}$	$3.76{\cdot}10^{-4}$	$2.20{\cdot}10^{-1}$
Shekel Fam. 5	$3.12{\cdot}10^{-1}$	$2.61{\cdot}10^{-1}$	$3.89{\cdot}10^{-1}$
Shekel Fam. 7	$2.95{\cdot}10^{-1}$	$2.95{\cdot}10^{-1}$	$3.77{\cdot}10^{-1}$
Shekel Fam. 10	$2.40{\cdot}10^{-1}$	$2.40{\cdot}10^{-1}$	$3.81{\cdot}10^{-1}$
Shekel's Foxholes	$6.70{\cdot}10^{-2}$	$6.70{\cdot}10^{-2}$	$4.87{\cdot}10^{-1}$
Six Hump	$2.65{\cdot}10^{-1}$	$2.32{\cdot}10^{-1}$	$1.40{\cdot}10^{-1}$

in table 4 where the DE algorithm outperforms the results of the CMA-ES on those functions where the attraction basins are spread over all the search space (see table 2 to check the maximum distance between attraction basins).

Another interesting feature to analyze in multimodal functions is the size of the optimum attraction basin, the size of the largest one and the relationship between them, i.e., if those attraction basins are the same or not. According to the size of the attraction basins, when it is smaller than a 10% of the search space, an exploration behavior is needed to achieve the optimum one and after get it exploit it until reach the optimum solution. For that reason, the DE outperforms the behavior of the CMA-ES on those functions with small optimum attraction basins. On the other hand, when the optimum attraction basin is large enough to be easily founded the CMA-ES results go beyond the DE results because of its exploitation capabilities. According to the relationship between the largest and the optimum attraction basins, when the optimum attraction basin is not the largest one a more explorative behavior is need. In this type of functions, like *Schwefel*, *Griewank*, *Hartman 6*, *Shekel Family 5* or *SixHump* functions, the algorithms tend to the largest attraction basins. The results of the CMA-ES confirms the fact that an exploitative behavior is not recommended. With this type of strategy, it is easier to reach the largest attraction basin than the optimum one. There is an exception, that also confirms the importance of the distance between attraction basins. *SixHump* function optimum attraction basin is smaller than the largest one but the distance between them is short enough to “jump” from one to the other and allowing the CMA-ES to obtain better results than the DE. The opposite behavior occurs on

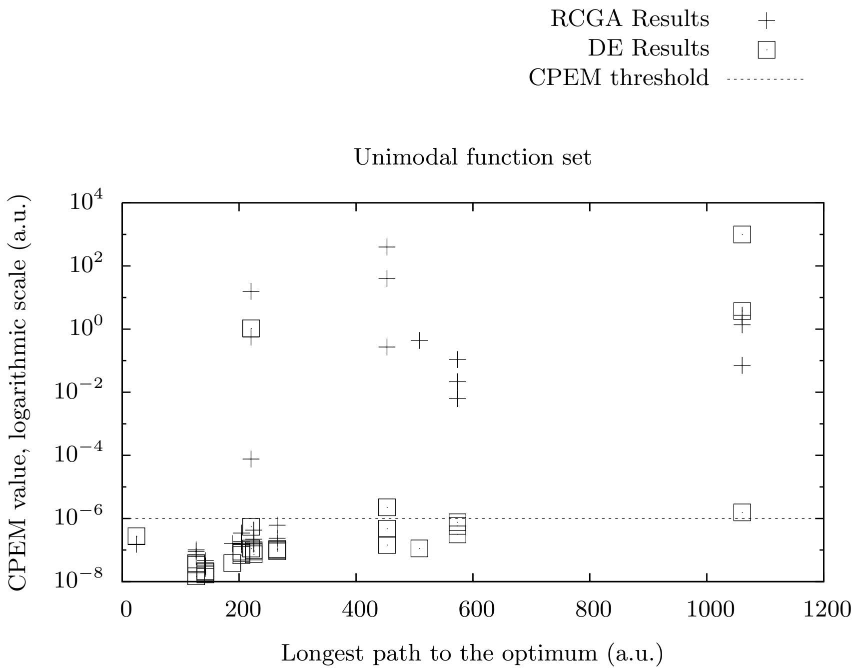
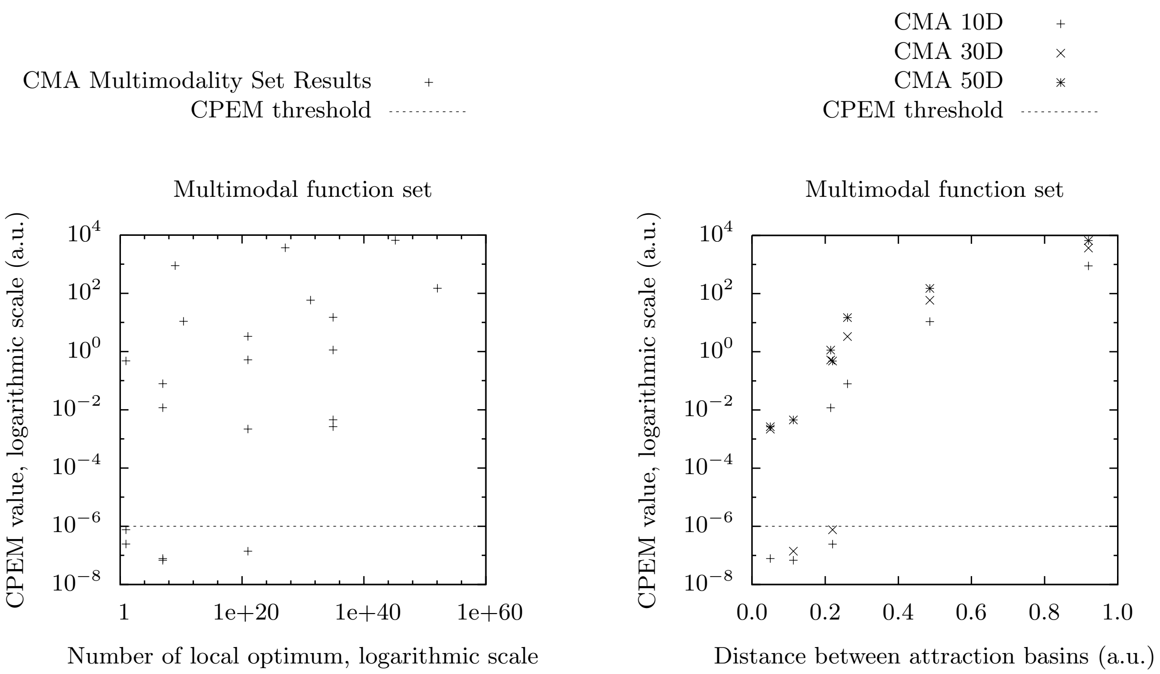


Figure 4: Results provided by the RCGA and DE algorithms over the unimodal functions subset characterized by the length of the longest path to the optimum.



(a) CMA-ES behavior according to the number of local optima.

(b) CMA-ES behavior according to the maximum distance between attraction basins.

Figure 5: Results provided by the CMA-ES algorithm in the multimodality functions subset.

the other functions where the distance is large and the DE outperforms the CMA-ES.

Finally, the behavior of the RCGA over multimodal functions is not related with the size of the attraction basins or the distance between them. Taking a look over the table 4 all the functions where the RCGA fails are Non-Separable, so we can conclude that its poor performance is more related with the separability than with the modality.

Summarizing, as we have shown in this section more information about the modality if the functions is needed in order to characterize their behavior. This extra information is need both in unimodal and multimodal functions. The path length in unimodal functions in useful to determine the difficulty of the functions, the longest the path the harder to find the solution to an EA without a strategy which adjust the mutation step automatically, like the RCGA implementation used here. About multimodal functions, the number of optima does not give enough information, function with a high number of them but not spread over the search space (like *Ackley's* functions) are easier to solve than function with few local optima but widely spread over the landscape (like *Hartman6* or *Shekel Family 5* functions). Moreover, the behavior of the EAs is highly dependent on the size of the attraction basins. Function where the optimum attraction basin is smaller are more difficult than those with very wide optimum attraction basin, without taking into account the number of them.

Table 3: Results provided by the three analyzed algorithms over the unimodal functions subset

Function	D	CPEM		
		RCGA	CMA-ES	DE
Axis Parallel-L	10	$1.55{\cdot}10^{-7}$	$2.11{\cdot}10^{-8}$	$7.34{\cdot}10^{-8}$
	30	$2.21{\cdot}10^{-7}$	$2.58{\cdot}10^{-8}$	$8.98{\cdot}10^{-8}$
	50	$4.27{\cdot}10^{-7}$	$3.03{\cdot}10^{-8}$	$9.88{\cdot}10^{-8}$
Schwefel 2.22-L	10	$1.66{\cdot}10^{-7}$	$3.27{\cdot}10^{-8}$	$9.19{\cdot}10^{-8}$
	30	$2.36{\cdot}10^{-7}$	$5.83{\cdot}10^{-8}$	$1.01{\cdot}10^{-7}$
	50	$6.10{\cdot}10^{-7}$	$1.09{\cdot}10^{-7}$	$1.08{\cdot}10^{-7}$
Sphere Model-L	10	$1.53{\cdot}10^{-7}$	$1.82{\cdot}10^{-8}$	$6.90{\cdot}10^{-8}$
	30	$1.85{\cdot}10^{-7}$	$1.76{\cdot}10^{-8}$	$8.08{\cdot}10^{-8}$
	50	$3.45{\cdot}10^{-7}$	$1.74{\cdot}10^{-8}$	$8.92{\cdot}10^{-8}$
Step-L	10	$6.29{\cdot}10^{-8}$	$9.92{\cdot}10^{-9}$	$1.47{\cdot}10^{-8}$
	30	$9.02{\cdot}10^{-8}$	$2.44{\cdot}10^{-8}$	$3.51{\cdot}10^{-8}$
	50	$1.01{\cdot}10^{-7}$	$4.99{\cdot}10^{-8}$	$3.88{\cdot}10^{-8}$
SumOf-L	10	$4.60{\cdot}10^{-8}$	$1.24{\cdot}10^{-7}$	$1.69{\cdot}10^{-8}$
	30	$3.10{\cdot}10^{-8}$	$2.89{\cdot}10^{-8}$	$2.13{\cdot}10^{-8}$
	50	$2.62{\cdot}10^{-8}$	$4.46{\cdot}10^{-8}$	$1.97{\cdot}10^{-8}$
Easom-NL	2	$1.52{\cdot}10^{-7}$	$1.95{\cdot}10^{-1}$	$2.72{\cdot}10^{-7}$
Schwefel 2.21-NL	10	$2.16{\cdot}10^{-2}$	$1.99{\cdot}10^{-7}$	$3.14{\cdot}10^{-7}$
	30	$6.29{\cdot}10^{-3}$	$9.88{\cdot}10^{1}$	$5.82{\cdot}10^{-7}$
	50	$1.08{\cdot}10^{-1}$	$1.00{\cdot}10^{2}$	$7.57{\cdot}10^{-7}$
Colville-N	4	$4.36{\cdot}10^{-1}$	$4.81{\cdot}10^{-8}$	$1.13{\cdot}10^{-7}$
Matyas-N	2	$1.60{\cdot}10^{-7}$	$1.40{\cdot}10^{-8}$	$3.86{\cdot}10^{-8}$
Perm-N	10	$7.80{\cdot}10^{-2}$	$8.40{\cdot}10^{-2}$	$1.56{\cdot}10^{-6}$
	30	$1.38{\cdot}10^{0}$	$1.61{\cdot}10^{1}$	$3.74{\cdot}10^{0}$
	50	$2.75{\cdot}10^{0}$	$2.26{\cdot}10^{1}$	$5.29{\cdot}10^{99}$
Schwefel 1.2-N	10	$2.71{\cdot}10^{-1}$	$2.74{\cdot}10^{-8}$	$1.44{\cdot}10^{-7}$
	30	$3.99{\cdot}10^{1}$	$5.08{\cdot}10^{-8}$	$4.73{\cdot}10^{-7}$
	50	$3.98{\cdot}10^{2}$	$7.19{\cdot}10^{-8}$	$2.26{\cdot}10^{-6}$
Zakharov (N)	10	$7.66{\cdot}10^{-5}$	$2.46{\cdot}10^{-8}$	$1.12{\cdot}10^{-7}$
	30	$5.63{\cdot}10^{-1}$	$5.52{\cdot}10^{-8}$	$5.47{\cdot}10^{-7}$
	50	$1.56{\cdot}10^{1}$	$8.99{\cdot}10^{-8}$	$1.06{\cdot}10^{0}$

4. CONCLUSIONS

This paper emphasizes the need of a more in depth analysis of the fitness landscapes in the functions typically used for benchmarking evolutionary algorithms with the objective of determining evolutionary algorithms with more precision what evolutionary algorithm is adequate (or is not appropriate) for what kind of function. This more in depth analysis is performed in this paper and information as the type of separability, the path length to the optimum and the size and distance between attraction basins is provided. This information allows a higher degree of knowledge regarding separability and modality, that are the most commonly features used in EAs competitions benchmark sets. In order to exemplify their usefulness, they have been applied to study the behavior of three well-known EAs showing that this deeper analysis allows to obtain reliable conclusions and relations between the search strategy of the EAs and their performance over different types of functions.

5. ACKNOWLEDGMENTS

APPENDIX

A. REFERENCES

[1] D. H. Ackley. *A connectionist machine for genetic hillclimbing.* Kluwer, Boston, 1987.

[2] E. Alba, B. Dorronsoro, M. Giacobini, and M. Tomassini. Decentralized cellular evolutionary algorithms. *Handbook of Bioinspired Algorithms and Applications*, 7:103–120, 2005.

[3] A. Auger and N. Hansen. A Restart CMA Evolution Strategy with Increasing Population Size. In *In Proceedings of the 2005 IEEE Congress on Evolutionary Computation, 2005.*, volume 2, pages 1769–1776, Apr 2005.

[4] P. Caamaño, A. Prieto, J. A. Becerra, F. Bellas, and R. J. Duro. Real-valued multimodal fitness landscape characterization for evolution. In *Proceedings of the 17th international conference on Neural information processing: theory and algorithms - Volume Part I*, ICONIP'10, pages 567–574, Berlin, Heidelberg, 2010. Springer-Verlag.

[5] P. Caamano, F. Bellas, J. A. Becerra, and R. J. Duro. Application domain study of evolutionary algorithms in optimization problems. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, GECCO '08, pages 495–502, New York, NY, USA, 2008. ACM.

[6] A. Eiben and M. Jelasity. A Critical Note on Experimental Research Methodology In EC. In *In Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002)*, pages 582–587. IEEE Press, 2002.

[7] J. Garnier and L. Kallel. How to detect all maxima of a function? In *Proceedings of the Second EVONET Summer School on Theoretical Aspects of Evolutionary Computing*, pages 343–370, 1999.

[8] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[9] F. Herrera, M. Lozano, and J. L. Verdegay. Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. *Artificial Intelligence Review*, 12:265–319, 1998.

[10] J. J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. N. Suganthan, C. A. C. Coello, and K. Deb. Problem Definitions and Evaluation Criteria

Table 4: Results provided by the three analyzed algorithms over the multimodal functions subset

Function	D	CPEM		
		RCGA	CMA-ES	DE
Aluffi-Pentini's-L	2	$6.64\cdot10^{-8}$	$2.57\cdot10^{-8}$	$3.94\cdot10^{-8}$
Becker and Lago-L	2	$7.51\cdot10^{-8}$	$1.61\cdot10^{-8}$	$5.06\cdot10^{-8}$
Bohach.1-L	2	$9.62\cdot10^{-8}$	$3.56\cdot10^{-8}$	$5.66\cdot10^{-8}$
Cosine Mixture-L	10	$9.28\cdot10^{-8}$	$1.18\cdot10^{-2}$	$4.98\cdot10^{-8}$
	30	$1.28\cdot10^{-7}$	$5.20\cdot10^{-1}$	$5.90\cdot10^{-8}$
	50	$2.04\cdot10^{-7}$	$1.41\cdot10^{0}$	$6.47\cdot10^{-8}$
Rastrigin-L	10	$6.15\cdot10^{-7}$	$1.09\cdot10^{1}$	$1.18\cdot10^{-7}$
	30	$6.29\cdot10^{-6}$	$5.86\cdot10^{1}$	$1.59\cdot10^{-7}$
	50	$5.38\cdot10^{-7}$	$1.50\cdot10^{2}$	$3.78\cdot10^{-7}$
Schwefel-L	10	$5.40\cdot10^{-2}$	$8.98\cdot10^{2}$	$1.66\cdot10^{-7}$
	30	$7.06\cdot10^{-5}$	$3.69\cdot10^{3}$	$3.27\cdot10^{-7}$
	50	$4.39\cdot10^{-7}$	$6.63\cdot10^{3}$	$5.70\cdot10^{-7}$
Ackleys-L	10	$2.21\cdot10^{-7}$	$5.31\cdot10^{-8}$	$1.10\cdot10^{-7}$
	30	$2.93\cdot10^{-7}$	$1.01\cdot10^{-7}$	$1.01\cdot10^{-7}$
	50	$7.67\cdot10^{-7}$	$1.49\cdot10^{-7}$	$1.49\cdot10^{-7}$
Griewank NL	30	$4.05\cdot10^{-7}$	$9.86\cdot10^{-4}$	$6.18\cdot10^{-8}$
	50	$4.50\cdot10^{-3}$	$1.91\cdot10^{-7}$	$9.29\cdot10^{-8}$
Levy-NL	10	$9.54\cdot10^{-8}$	$7.92\cdot10^{-2}$	$5.03\cdot10^{-8}$
	30	$1.41\cdot10^{-7}$	$3.33\cdot10^{0}$	$6.18\cdot10^{-8}$
	50	$2.18\cdot10^{-7}$	$1.50\cdot10^{1}$	$6.95\cdot10^{-8}$
Penalized1 NL	10	$1.34\cdot10^{-7}$	$6.81\cdot10^{-8}$	$6.00\cdot10^{-8}$
	30	$1.87\cdot10^{-7}$	$1.40\cdot10^{-7}$	$6.98\cdot10^{-8}$
	50	$2.58\cdot10^{-7}$	$4.52\cdot10^{-3}$	$7.76\cdot10^{-8}$
Penalized2 NL	10	$2.18\cdot10^{-7}$	$7.81\cdot10^{-8}$	$6.27\cdot10^{-8}$
	30	$2.42\cdot10^{-7}$	$2.20\cdot10^{-3}$	$1.21\cdot10^{-7}$
	50	$3.39\cdot10^{-7}$	$2.64\cdot10^{-3}$	$8.51\cdot10^{-8}$
Beale-N	2	$2.30\cdot10^{-7}$	$3.77\cdot10^{-8}$	$4.70\cdot10^{-8}$
Bohach.2-N	2	$9.85\cdot10^{-8}$	$3.52\cdot10^{-8}$	$6.26\cdot10^{-8}$
Dekkers & Aarts-N	2	$4.38\cdot10^{-8}$	$1.41\cdot10^{-8}$	$1.85\cdot10^{-8}$
Goldstein Price-N	2	$1.13\cdot10^{-7}$	$3.01\cdot10^{-8}$	$5.25\cdot10^{-8}$
Griewank-N	10	$1.03\cdot10^{-2}$	$3.20\cdot10^{-2}$	$3.93\cdot10^{-7}$
Hart3-N	3	$3.77\cdot10^{-8}$	$1.71\cdot10^{-8}$	$2.94\cdot10^{-8}$
Hart6-N	6	$1.01\cdot10^{-2}$	$5.30\cdot10^{-3}$	$1.06\cdot10^{-1}$
Rosenbrock (N)	10	$5.28\cdot10^{0}$	$2.45\cdot10^{-7}$	$6.47\cdot10^{-7}$
	30	$2.38\cdot10^{1}$	$7.60\cdot10^{-7}$	$8.48\cdot10^{-1}$
	50	$3.03\cdot10^{1}$	$4.78\cdot10^{-1}$	$3.67\cdot10^{1}$
Kowalik-N	4	$5.13\cdot10^{-4}$	$1.32\cdot10^{-4}$	$3.14\cdot10^{-4}$
ShekF5-N	4	$2.09\cdot10^{0}$	$7.03\cdot10^{-1}$	$2.07\cdot10^{-7}$
ShekF7-N	4	$9.66\cdot10^{-1}$	$1.33\cdot10^{-7}$	$1.35\cdot10^{-7}$
ShekF10 N	2	$2.68\cdot10^{-1}$	$1.34\cdot10^{-7}$	$1.72\cdot10^{-7}$
Shekel's Foxholes-N	2	$1.19\cdot10^{-1}$	$5.31\cdot10^{-7}$	$1.17\cdot10^{-7}$
Six Hump-N	2	$5.57\cdot10^{-8}$	$1.59\cdot10^{-8}$	$5.02\cdot10^{-8}$

for the CEC 2006 Special Session on Constrained Real-Parameter Optimization. Technical report, Nanyang Technological University, 2006.

[11] M. Locatelli. A note on the Griewank test function. *Journal of Global Optimization*, 25(2):169–174, 2003.

[12] D. Mosk-Aoyama and D. Shah. Computing separable functions via gossip. In *PODC '06: Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 113–122, New York, NY, USA, 2006. ACM.

[13] A. Nimtawat and P. Nanakorn. Automated layout design of beam-slab floors using a genetic algorithm. *Computers and Structures*, 87(21-22):1308 – 1330, 2009.

[14] G. Rawlins. *Foundations of Genetic Algorithms*. Morgan Kaufmann, 1991.

[15] J. Rönkkönen, S. Kukkonen, and K. V. Price. Real-Parameter Optimization with Differential Evolution. In *The 2005 IEEE International Congress on Evolutionary Computation (CEC'05)*, page 12, Apr 2005.

[16] M. M. Stickberger. *Genetics*. Collier-McMillan Ltd, 1968.

[17] R. Storn and K. V. Price. Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11:341 – 359, Nov 1997.

[18] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari. Problem definitions and evaluation criteria for the CEC 2005 Special Session on Real Parameter Optimization. Technical report, Nanyang Technological University, 2005.

[19] H. Tang, S. Xue, and C. Fan. Differential evolution strategy for structural system identification. *Computers and Structures*, 86(21-22):2004 – 2012, 2008.

[20] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise. Benchmark Functions for the CEC'2010 Special Session and Competition on Large-Scale Global Optimization. Technical report, University of Science and Technology of China (USTC), School of Computer Science and Technology, Nature Inspired Computation and Applications Laboratory (NICAL): Héféi, Ānhuī, China, Jan. 8, 2010.

[21] K. Tang, X. Yao, P. Suganthan, C. Macnish, Y. Chen, C. Chen, and Z. Yang. Benchmark Functions for the CEC 2008 Special Session and Competition on Large Scale Global Optimization. Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, China, November 2007.

[22] K. Tang, X. Yao, P. Suganthan, C. MacNish, Y. Chen, C. Chen, and Z. Yang. Benchmark functions for the CEC2́008 special session and competition on large scale global optimization. *Nature Inspired Computation and Applications Laboratory, USTC, China, Tech. Rep*, 2007.

[23] M.-H. Tseng and H.-C. Liao. The genetic algorithm for breast tumor diagnosis–The case of DNA viruses. *Applied Soft Computing*, 9(2):703 – 710, 2009.

[24] L. D. Whitley, K. E. Mathias, S. B. Rana, and J. Dzubera. Building Better Test Functions. In *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 239–247, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.