Information Sciences 289 (2014) 41–58

Contents lists available at ScienceDirect

Information Sciences

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

Analyzing convergence performance of evolutionary algorithms: A statistical approach

Joaquín Derrac [a], Salvador García [b], Sheldon Hui [c], Ponnuthurai Nagaratnam Suganthan [c,*], Francisco Herrera [d,e]

[a] School of Computer Science & Informatics, Cardiff University, Cardiff CF24 3AA, United Kingdom
[b] Department of Computer Science, University of Jaén, 23071 Jaén, Spain
[c] School of Electrical and Electronic Engineering, Nanyang Technological University, 50 Nanyang Avenue, 639798 Nanyang, Singapore
[d] Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain
[e] Faculty of Computing and Information Technology – North Jeddah, King Abdulaziz University, 21589 Jeddah, Saudi Arabia

ARTICLE INFO

Article history:
Received 21 November 2013
Received in revised form 21 April 2014
Accepted 1 June 2014
Available online 8 August 2014

Keywords:
Page's trend test
Nonparametric tests
Convergence-based algorithmic comparison
Evolutionary algorithms

ABSTRACT

The analysis of the performance of different approaches is a staple concern in the design of Computational Intelligence experiments. Any proper analysis of evolutionary optimization algorithms should incorporate a full set of benchmark problems and state-of-the-art comparison algorithms. For the sake of rigor, such an analysis may be completed with the use of statistical procedures, supporting the conclusions drawn.

In this paper, we point out that these conclusions are usually limited to the final results, whereas intermediate results are seldom considered. We propose a new methodology for comparing evolutionary algorithms' convergence capabilities, based on the use of Page's trend test. The methodology is presented with a case of use, incorporating real results from selected techniques of a recent special issue. The possible applications of the method are highlighted, particularly in those cases in which the final results do not enable a clear evaluation of the differences among several evolutionary techniques.

1. Introduction

An analysis based on final results is the most popular way in which the performance of Computational Intelligence search methods is assessed. For example, in the field of evolutionary optimization, algorithms are usually evaluated with respect to the quality of the best result obtained, over a predefined set of benchmark functions. However, there are other traits of evolutionary algorithms that are worthy of analysis, beyond the quality of the final solution reached: Efficiency, applicability to different domains, diversity management and convergence [2].

Convergence is usually acknowledged to be a desirable capability for every new search algorithm designed today. In the case of Evolutionary Algorithms (EAs), this is a staple concern in the sense that good convergence is a must-have for any new technique to be accepted by the research community [4,7,30]. However, it is common to see convergence analyzed only as the *capability of the technique to reach the final*, regardless of how quickly such a result is reached.

* Corresponding author. Tel.: +65 6790 5404; fax: +65 6793 3318.
E-mail addresses: jderrac@decsai.ugr.es (J. Derrac), sglopez@ujaen.es (S. García), SHUI1@e.ntu.edu.sg (S. Hui), epnsugan@ntu.edu.sg (P.N. Suganthan), herrera@decsai.ugr.es (F. Herrera).

http://dx.doi.org/10.1016/j.ins.2014.06.009
0020-0255/

In this sense, the development of a methodology to assess the convergence performance of several algorithms – that is, which algorithm converges faster – is important, particularly in cases in which a benchmark problem is unable to differentiate algorithms using the final results achieved.

The conclusions obtained after analyzing the final results of the algorithms are often backed up by using statistical techniques. Nonparametric tests [8,22] are preferred for this task due to the absence of strong limitations regarding the kind of data to analyze (in contrast with parametric tests, for which the assumptions of normality, independence and homoscedasticity of the data are necessary for the sake of reliability) [18,15,31,41].

Throughout this paper, we show how Page's trend statistical test [27] can be applied to the analysis of pairwise convergence. It is a nonparametric test for multiple classification, which allows trends to be detected among the results of the treatments if the null hypothesis of equality is rejected. In our case, if the treatments are chosen as the differences between the fitness values of two algorithms, computed at several points of the run (*cut-points*), the test can be used to detect increasing and decreasing trends in the differences as the search goes on. The study of these trends, representing the evolution of the algorithms during the search, enables us to develop a new methodology for comparing algorithms' convergence performance.

The description of our approach is completed with the inclusion of an alternative version for computing the ranks of the test. This second version allows the test to be applied safely should one of the algorithms reach the optimum of some of the benchmark functions before the end of the run (which would prevent it from progressing further, thereby preventing the proper evaluation of its convergence in the last stages of the search).

To demonstrate the usefulness of both the basic and the alternative versions of the test, a full case study is presented. The study compares the performance of several EAs for continuous optimization, namely advanced versions of the Differential Evolution evolutionary technique [32,11]. It is based on the submissions accepted for the Special Issue on Scalability of Evolutionary Algorithms and other Metaheuristics for Large Scale Continuous Optimization Problems [21] in the Soft Computing journal.

As will be shown in the study, the use of Page's trend test can be very useful when analyzing the performance of the algorithms throughout the search. Its use provides the researchers with a new perspective for assessing how the algorithms behave, considering intermediate results instead of just the final results in each function. This can reveal very illustrative information when comparing the methods, particularly in cases where the final results are statistically similar.

A further contribution presented in this work is the development of a Java program to implement our approach. The program processes the intermediate results of two or more algorithms. After that, Page's trend test is carried out for every pair of algorithms, and the results are output in TeX format. It can be downloaded at the following URL: http://sci2s.ugr.es/sicidm/pageTest.zip.

The rest of this paper is organized as follows: Section 2 provides some background regarding the use of nonparametric tests to contrast the results of evolutionary optimization experiments. Section 3 presents our approach, detailing how Page's trend test can be applied to compare the convergence performance of two algorithms. Section 4 describes the case study chosen to illustrate the application of the test. Section 5 presents the results obtained and the related discussions. Section 6 concludes the paper. Three appendices are also included, respectively providing a guide to obtaining and using the software used to run the test (A), detailed final results of the case of study (B) and the full results of the application of Page's trend test (C).

2. Background

The assessment of the performance of algorithms is an important task when performing experiments in Computational Intelligence. When comparing EAs, it is necessary to consider the extent to which the *No Free Lunch* theorem [39] limits the conclusions: Under no specific knowledge, any two algorithms are equivalent when their performance is averaged across all possible problems.

Therefore, assuming that EAs take advantage of the available knowledge in one way or another, it is advisable to focus interest on efficiency and/or effectiveness criteria. When theoretical developments are not available to check such criteria, the analysis of empirical results can help to discern which techniques perform more favorably for a given set of problems.

In the literature, it is possible to find different viewpoints on how to improve the analysis of experiments [23]: The design of test problems [13] (for example, the design of complex test functions for continuous optimization [14,38]), the use of advanced experimental design methodologies (for example, methodologies for adjusting the parameters of the algorithms depending on the settings used and results obtained [1,2] or for performing Exploratory Landscape Analysis [3,26]) or the analysis of the results [9] (to determine whether the differences between algorithms' performances are significant or not). Another example is [35], where a method inspired in chess rating systems is adapted to rank the performance of evolutionary algorithms.

From the statistical analysis perspective, the use of statistical tests enhances the conclusions drawn, by determining whether there is enough *evidence* to reject null hypotheses based on the results of the experiments. For this task, it is possible to find applications of both parametric [29,10] and, more recently, nonparametric [18,24,12] statistical procedures.

Nonparametric tests are used to compare algorithms' final results, represented as average values for each problem (using the same criterion: average, median, etc. over the same number of runs for each algorithm and problem). This usually enables practitioners to rank differences among algorithms and determine which ones are significant, thus leading to a characterization of which algorithms behave better than the rest.

However, a drawback of this methodology is that it only takes into consideration the final results obtained at the end. When analyzing EAs, this often overshadows interesting conclusions which could be drawn by analyzing the performance of the algorithms during the whole run.

The rest of this section is devoted to the introduction of nonparametric tests and the classical definition of Page's trend test. This provides the necessary background to present our proposal on the use of nonparametric tests to analyze the convergence performance of EAs, as an enhancement to the final-results oriented statistical analysis, in particular when final results are statistically similar.

2.1. Nonparametric tests

Nonparametric tests [19] are powerful tools for the analysis of results in Computational Intelligence. They can be used to analyze both nominal and real data, through the use of rank-based measures. At the cost of some inference power (when compared with their parametric counterparts), they offer safe and reliable procedures to contrast the differences between different techniques, particularly in multiple-problem analysis (that is, for studies in which the results over multiple problems are analyzed jointly, instead of performing a single test per each problem).

To apply nonparametric tests to a multiple-problem set-up, a result per algorithm/problem pair must be provided. This is often obtained as the average of a given performance measure over several runs – carried out on every single problem. A typical example could be the average error on 50 runs of an algorithm over 25 different benchmark problems.

A null or no-effect hypothesis is to be formulated prior to the application of the test. It often supports the equality or absence of differences among the results of the algorithms, and enables alternative hypotheses to be raised that support the opposite [31]. The null hypothesis can be represented by H_0, and the alternative hypotheses by $H_1, \ldots, H_n$. The application of the tests leads to the computation of a statistic, which can be used to reject the null hyphotesis at a given level of significance α.

For a fine grained analysis, it is also possible to compute the smallest level of significance that results in the rejection of the null hypothesis. This level is the p-value, which is the probability of obtaining a result at least as extreme as the one that was actually observed, assuming that the null hypothesis is true. The use of p-values is often preferred over using only fixed α levels since they provide cleaner measures of how significant the result is (the smaller the p-value, the stronger the evidence against the null hypothesis is) [41].

The nonparametric tests can be classified by their capabilities to perform pairwise comparisons and multiple comparisons. It is important to note that the p-values obtained through pairwise comparisons are independent, and thus multiple comparison procedures should be used instead when comparing more than two algorithms [17].

Several nonparametric tests can be used to compare the final results of EAs in continuous optimization problems: The Sign test and the Wilcoxon Signed-ranks test can help dealing with pairwise comparisons, whereas the Friedman, the Friedman Aligned-ranks and the Quade test can be used for performing multiple comparisons. Post-hoc procedures, such as the Holm test can be introduced after the application of multiple comparisons, to characterize the existence of pairwise differences within a multiple comparisons set-up [16].

2.2. Page's trend test

Page's trend test for ordered alternatives [27] can be classified in the family of tests for association in multiple classifications, similar to the Friedman test. Before detailing its application to the analysis of convergence performance (which will be provided in the next section), it is necessary to provide its original definition.

This test defines the null hypothesis as the equality between the k treatments analyzed that can be rejected in favor of an *ordered* alternative (the *ordered* alternative is the main difference of this test with respect to the Friedman test, which only defines the alternative hypothesis as the existence of differences between treatments).

The ordered alternative must be defined by the practitioner before starting the analysis. An order between the k treatments has to be provided, and it should reflect the expected order for the populations. Hence, the treatments' measures should be numbered from 1 to k, where treatment 1 has the smallest sum of ranks, and treatment k has the largest.

Once such an order and the data (consisting of n samples of the k treatments) are provided, the n samples (data rows) can be ranked from the best to the worst, giving a rank of 1 to the best measure in the sample, a rank of 2 to the second, . . ., and a rank of k to the worst. If there are ties for a given sample, average ranks can be assigned (for example, a tie between the first and the second result would produce an average rank of $(1 + 2)/2 = 1.5$, which would be assigned to both measures). If the data is consistent with the initial ordering defined, then the sum of ranks' values for each of the treatments will follow in increasing order.

After obtaining the ranks, the Page L statistic can be computed using the following expression

$$L = \sum_{j=1}^{k} jR_j = R_1 + 2R_2 + \cdots + kR_k \tag{1}$$

where $R_j = \sum_{i=1}^{n} r_i^j$, and r_i^j is the rank of the j-th of k measures on the i-th of n samples.

The L statistic can be seen as a weighted version of Friedman's test (as presented in [27]) by which average ranks are given more weight the closer they are to the final treatments. L critical values can be computed for small values of k and n (see, for example, Table Q in [19] for values up to $k = 8$ and $n = 12$). In the case that larger values are required, a normal approximation should be considered. The normal approximation for the L statistic is given by the following expression

$$Z = \frac{12(L - 0.5) - 3Nk(k+1)^2}{k(k+1)\sqrt{N(k-1)}}. \tag{2}$$

Table 1
Computation of ranks for Page's trend test (Example 1).

Ranks	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
Function 1	1	2	3	10	6.5	6.5	6.5	6.5	6.5	6.5
Function 2	10	4	9	8	7	6	5	3	2	1
Function 3	1	2	3	4	5	6	10	9	8	7
Function 4	9	10	8	7	6	5	4	3	2	1
Function 5	1	2	3	10	6.5	6.5	6.5	6.5	6.5	6.5
Function 6	1	2	3	4	10	9	6.5	6.5	6.5	6.5
Function 7	10	9	8	7	6	5	2.5	2.5	2.5	2.5
Function 8	10	9	8	7	6	5	4	3	2	1
Function 9	1	2	3	4	5	6	10	9	8	7
Function 10	1	2	3	4	10	7	7	7	7	7
Function 11	1	2	3	4	5	6	10	9	8	7
Function 12	1	10	9	8	7	6	5	4	3	2
Function 13	1	2	3	4	5	6	10	9	8	7
Function 14	9	10	8	7	6	5	4	3	2	1
Function 15	1	2	3	4	5	6	8.5	8.5	8.5	8.5
Function 16	10	9	8	7	6	5	4	3	2	1
Function 17	1	2	3	4	5	6	10	9	8	7
Function 18	9	10	2	1	3	4	8	7	6	5
Function 19	1	2	3	4	5	8	8	8	8	8

whose estimation, including a continuity correction, is approximately standard normal with a rejection region on the right tail.

3. Using Page's trend test for convergence analysis

In this section, the use of the Page's trend test for convergence analysis is described. The test is applied under the assumption that an algorithm with a good convergence performance will advance towards the optimum faster than another algorithm with a worse performance. Thus, differences in the fitness values will increase as the search continues.

The application of Page's trend test to this task is described in Section 3.1. A modification to the ranks assignment procedure of the test (and hence, to this proposal) is presented in Section 3.2. This modification, useful in cases where the algorithms reach the optimum of some functions before the end of the experiments, may be of interest for dealing with many common experimental studies on continuous optimization, whose functions' optima are very likely to be reached for some of the functions of the benchmarks.

3.1. Using Page's trend test

The original definition of Page's trend test focuses on detecting increasing trends in the rankings computed using the input data. This means that decreasing trends in the data values will be detected, provided that ranks are computed as described before.

The input data would represent the differences between each algorithm's average best objective value reached, at different steps of the search (*cut-points*). The best objective value reached at each cut-point has to be collected for every run of each algorithm and function. These values should be then averaged along the runs, so that a single, aggregated value is obtained per each algorithm, function and cut-point. This will allow us to compute the differences between a pair of algorithms, by subtracting the aggregated values.

Therefore, the input data of the test will represent the differences between the two algorithms, **A** and **B**, recorded at c different points of the search, on n problems (functions).

The treatments (columns) will represent each of the c *cut-points* at which data is gathered (they should be taken at regular intervals), whereas the samples (rows) will represent the n different functions used to test the algorithms. Fig. 1 shows an example of the convergence of two algorithms and how $c = 10$ cut-points are tracked for each one.

The specific number of samples and treatments to consider would depend on the characteristics of each specific situation and the available data, although a reasonable rule would be to have approximately twice the number of samples as treatments, at least (see [19]). Also, treatments should always be ordered in increasing order, since we are interested in analyzing the trends as the search progresses. That is, the first treatment should represent the first cut-point, the second treatment should represent the second cut-point and so on.

Under these conditions, Page's trend test may be used to detect increasing trends in the ranks that represent the differences (or decreasing trends, if the order of the algorithms is reversed). Assuming a minimization objective[1], the outcome of the test can be interpreted as follows:

[1] The test could be easily adapted to work with maximization objectives, by reversing the sign of the differences.

- Significant increasing trend: If a consistently increasing trend in the ranks is found, this means that either the fitness of **A** is growing faster than the fitness of **B** or that the fitness of **B** is decreasing faster than the fitness of **A**.
  Since the fitness is computed as the best value found throughout the search, the former case is impossible. Hence, if an increasing trend is detected, this means that the fitness of **B** is decreasing faster, which means that it has a better convergence performance.
- Significant decreasing trend: Following the same reasoning as above, this could only mean that the fitness of **A** is decreasing faster. Hence, a decreasing trend in the ranks means that **A** has a better convergence performance.
- No significant trend: If no consistent trend is found, then nothing can be said about the relative convergence performance of two algorithms.

Example 1. Let **A** and **B** be two algorithms to analyze, considering $n = 19$ different functions and $c = 10$ cut-points. Table 1 shows an example of the treatments' ranks (R_j) computed for the **A–B** differences in fitness values. Note that ranks are assigned from 1 (greater absolute differences) to 10 (lower absolute differences), and that midranks are assigned when necessary (hence, the sum of all the R_j values will always be 55).

Fig. 2 shows the sum of all the R_j values per cut-point, and the Page's L statistic computed from them. It shows the associated p-value obtained. For completness, the relevant data of the opposite comparison (**B–A**) is also included.

The comparison **A–B** shows an increasing trend in the ranks (as can be seen in the figure), which is confirmed by a very low p-value. Moreover, the opposite comparison **B–A**, shows clearly that the ranks are not increasing (in fact, they are decreasing), which is rejected by a p-value near to 1.0. These results show that the algorithm **A** is converging faster than algorithm **B**.

3.2. Alternative ranks computation procedure

Although the aforementioned procedure should be correct in most cases, it should be used with caution if any of the algorithms is unable to progress in the search for some of the functions. The most typical case of this occurring would probably

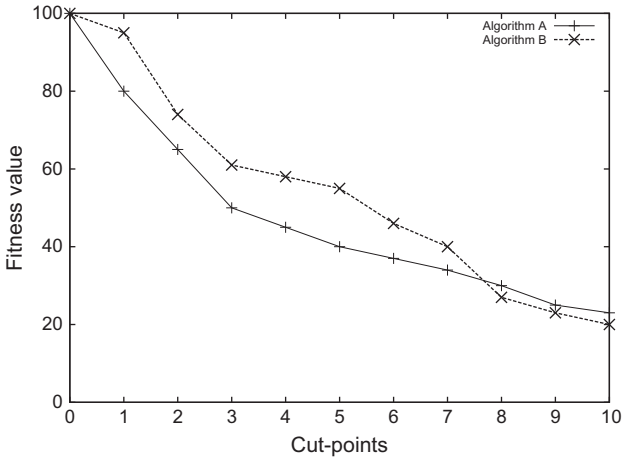

Fig. 1. Convergence behavior of two algorithms at 10 different cut-points.

Differences	A - B	B - A
C1	82.0	127.0
C2	81.0	128.0
C3	85.0	124.0
C4	110.0	99.0
C5	127.0	82.0
C6	114.0	95.0
C7	105.5	103.5
C8	113.5	95.5
C9	113.5	95.5
C10	113.5	95.5
L	6061.0	5434.0
p-value	0.00451	0.99560

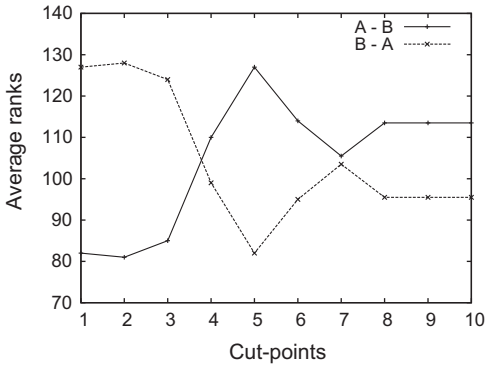

Fig. 2. Page's trend test results (Example 1).

be when the absolute global optimum could be reached within the evaluations limit stipulated (something that it is very likely to happen with most of the common benchmarks currently used, such as the IEEE Conference on Evolutionary Computation 2005 Special Session on Real-Parameter Optimization [33] one). Hence, if such optimum is reached (if it is known), the computation procedure should be corrected in order rank the difference properly.

Fig. 3 shows a graph depicting the convergence process of two different algorithms for a given function. As can be seen, algorithm **A** is converging faster than algorithm **B**, reaching the optimum using half of the total evaluations allowed. In this situation, the test would be expected to report a positive response, showing that algorithm **A** has a better behavior in that problem.

However, this is not the case if the former ranking computation procedure is used. If fitness value differences between the algorithms are computed, an increasing trend would be identified from cut-points 1 to 5 (positive for algorithm **A**, since the differences are increasing as the algorithms proceed). However, a decreasing trend would also be detected, from cut-points 6 to 10. Clearly, this undesirable behavior is caused by the fact that the optimum has been reached too soon by algorithm **A**, preventing it from progressing further for the rest of the fitness evaluations.

To tackle this problem, we propose a modification to the procedure used to compute the ranks. The aim of this alternative version is to continue using the same ranking scheme as in the original approach, but fixing the ranks for those cases in which the function's optimum is reached well before the maximum number of fitness evaluations are exhausted.

When analyzing the differences between two algorithms, **A** and **B**, as **A–B**, four different cases can be highlighted (by considering ranks for 10 cut-points for each case):

1. **No algorithm reaches the optimum before the end**: No further changes are necessary.
   Example: Using 10 cut-points, a possible ordering could be the following:

   $$7, 3, 5, 2, 1, 4, 8, 6, 9, 10$$

(Ranks are computed in the standard way. That is, the difference at the first cut-point is the seventh largest absolute difference (rank 7), the difference at the second one is the third largest absolute difference (rank 3), and so forth. The largest difference is found at the fifth cut-point (rank 1), whereas the smallest is found at the last cut-point (rank 10)).

2. **Algorithm A reaches the optimum before the end**: Ranks should be modified so an increasing trend is detected from the point at which algorithm **A** reaches the optimum to the last cut-points of the comparison. The rest of the ranks could be assigned as above.
   Example: Using 10 cut-points and having algorithm **A** ending at the 6th cut-point, a possible ordering could be the following:

   $$3, 1, 2, 4, 5, \mathbf{6}, \mathbf{7}, \mathbf{8}, \mathbf{9}, \mathbf{10}.$$

(Highest ranks are assigned increasingly starting from the sixth cut-point).

3. **Algorithm B reaches the optimum before the end**: Ranks should be modified so a decreasing trend is detected from the point at which algorithm **B** reaches the optimum. The lowest ranks should be assigned decreasingly to the last cut-points of the comparison. The rest of the ranks could be assigned as in the first case.
   Example: Using 10 cut-points and having algorithm **B** converging to the global optimum in the 6th cut-point, a possible ordering could be the following:

   $$6, 10, 8, 9, 7, \mathbf{5}, \mathbf{4}, \mathbf{3}, \mathbf{2}, \mathbf{1}.$$

(Lowest ranks are assigned decreasingly starting from the sixth cut-point).

4. **Both algorithms reach the optimum at the same cut-point**: In this case, the computation of the ranks can be performed as with the original version of the test. Zero differences will be ranked using the median ranks, denoting that no trend is detected from the point in which both algorithms reached the optimum.
   Example: Using 10 cut-points and having algorithms **A** and **B** ending in the sixth cut-point, a possible ordering could be the following:

   $$1, 10, 9, 8, 2, \mathbf{5}, \mathbf{5}, \mathbf{5}, \mathbf{5}, \mathbf{5}.$$

(Assigning midranks from the sixth cut-point).

As has been shown, the alternative version addresses those cases in which the global optimum is reached prior to exhausting the maximum function evaluations, preventing the algorithm from continuing with the search. In cases 2 and 3, the scheme adopted will benefit the first algorithm to converge to the optimum, and this benefit will be higher the sooner the said optimum is reached. Also, note that if case 4 is reached, the chances of not rejecting the equality hypothesis will greatly increase (as would happen naturally if the basic ranking scheme is considered).

Example 2. Table 2 show the cutpoint at which algorithms **A** and **B** (from Example 1) reached the known optimum, of 19 different functions.

As can be seen in the table, algorithm **B** always reaches the optimum at the same time or before the algorithm **A** (except in function F7). Hence, it would be expected that **B** would be found to have a better convergence behavior.

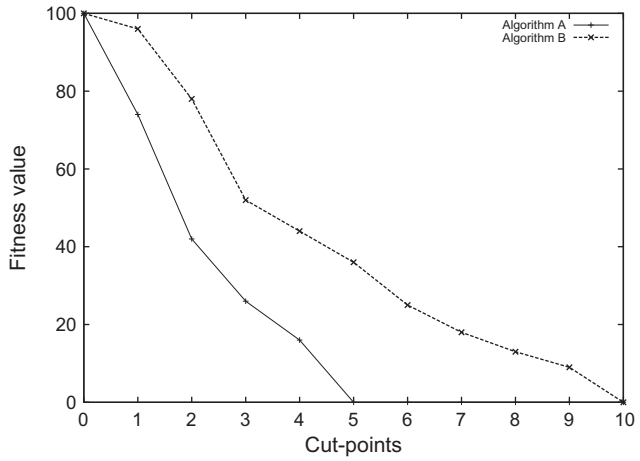

Fig. 3. Convergence behavior of two algorithms at 10 different cut-points. Algorithm **A** converges faster than algorithm **B**.

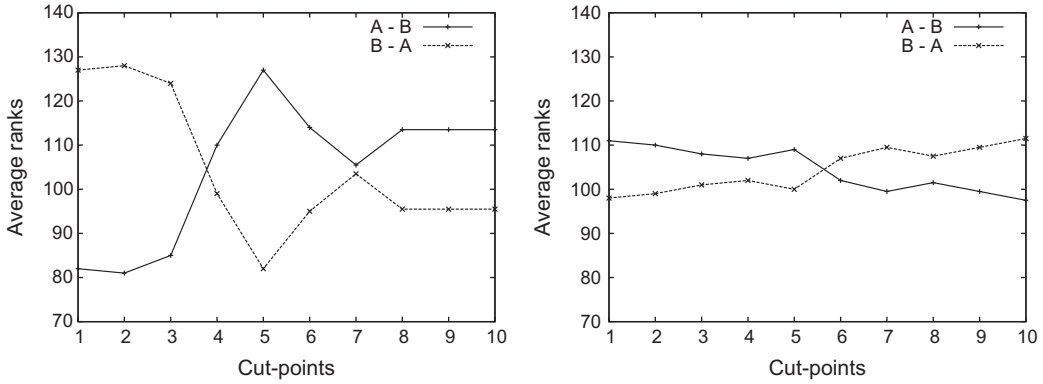

Fig. 4. Ranks computed in the Example 2.

Table 2
Ending cut-points for algorithms A and B (Example 2. "–" denotes that the optimum was not reached.

Ending	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19
Algorithm A	5	–	–	10	5	7	7	–	–	6	–	10	–	–	7	–	–	–	6
Algorithm B	4	–	–	5	4	5	–	–	–	5	–	5	–	–	7	3	–	–	6

Table 3 shows the ranks, L statistic and p-values computed using this data, by both versions of Page's trend test (standard and alternative). Note that the results of the first two rows are the same as were shown in Example 1. These results are also depicted graphically in Fig. 4

The alternative computation of the ranks introduces a clear modification in the *p*-values computed. Such modification corrects the previous result, in the sense that a favourable trend is identified this time for algorithm **B**. This is in consonance with the data shown in Table 2, clearly depicting algorithm **B** as the one with the best convergence performance.

4. Case of use: On analyzing the performance of several Differential Evolution based approaches

The proposed methodology should make it possible to carry out pairwise comparisons among algorithms involved in an experimental study. At this point, our objective is to show what could be the outcome of such comparisons and how to interpret them.

Hence, in order to demonstrate the usefulness of the methodology proposed, the following sections will be devoted to describing a case study focused on the analysis of several Differential Evolution [11] based techniques, and to analyzing the results obtained. It is complemented by the description of a software developed to apply the test (in A), which can be obtained at http://sci2s.ugr.es/sicidm/pageTest.zip.

This case is mainly seeded on the Soft Computing journal Special Issue on Scalability of Evolutionary Algorithms and other Metaheuristics for Large Scale Continuous Optimization Problems [21] from which both the benchmarking functions and

Table 3
Original and alternative version for the computation of ranks (Example 2).

Differences	Original **A–B**	Original **B–A**	Alternative **A–B**	Alternative **B–A**
C1	79	130	111	98
C2	93	116	118	91
C3	93	116	112	97
C4	108	101	102	107
C5	115	94	100	109
C6	114	95	105	104
C7	129.5	79.5	122.5	86.5
C8	116.5	92.5	106.5	102.5
C9	104.5	104.5	91.5	117.5
C10	92.5	116.5	76.5	132.5
L	5939	5556	5519	5976
p-Value	0.05554	0.94539	0.97196	0.02858

some of the participant algorithms have been taken. Note that, even in large case studies like this one, this methodology requires to consider only a fixed number of cut-points (dependent on the number of functions considered as benchmark). This requirement is independent to other factors like population sizes or the number of iterations performed, making the test a suitable choice when analyzing complex experiments.

4.1. Functions

The benchmark proposed in the special issue consists of 19 functions [20]. The first six were taken from the CEC'2008 Special Session and Competition on Large Scale Global Optimization [34]. Functions F7–F11 were included as shifted versions from other common benchmarks in continuous optimization. Finally, functions F12–F19 were built for this benchmark combining two of the previous ones (at least one of the functions in each combination is non-separable). Table 4 shows the main characteristics of each function: **Name**, **Range** and **Optimum value**.

These functions were presented [21] as a suitable benchmark for testing the capabilities of EAs and other metaheuristics. By including unimodal/multimodal, separable/non-separable and shifted functions, this benchmark should pose a challenge for modern optimization algorithms.

All the 19 functions will be considered for the study. The analysis of results will be carried out considering three different set-ups: 50 dimensions, 100 dimensions and 200 dimensions. This will provide a clear picture on how the algorithms perform as the dimensionality of the problems increase.

4.2. Algorithms considered

Six different algorithms have been chosen for this study, 5 of which where originally accepted for the special issue [21]. All of them are advanced EAs based on differential evolution:

Table 4
The 19 test functions chosen as benchmark.

Function	Name	Range	Optimum
F1	Shifted Sphere Function	$[-100, 100]^D$	−450
F2	Shifted Schwefel's Problem 2.21	$[-100, 100]^D$	−450
F3	Shifted Rosenbrock's Function	$[-100, 100]^D$	390
F4	Shifted Rastrigin's Function	$[-5, 5]^D$	−330
F5	Shifted Griewank's Function	$[-600, 600]^D$	−180
F6	Shifted Ackley's Function	$[-32, 32]^D$	−140
F7	Shifted Schwefel's Problem 1.2	$[-65.536, 65.536]^D$	0
F9	Shifted Extended $f10$	$[-100, 100]^D$	0
F10	Shifted Bohachevsky	$[-15, 15]^D$	0
F11	Shifted Schaffer	$[-100, 100]^D$	0
F12	Composite F9 + F1 Function	$[-100, 100]^D$	0
F13	Composite F9 + F3 Function	$[-100, 100]^D$	0
F14	Composite F9 + F4 Function	$[-5, 5]^D$	0
F15	Composite F10 + F7 Function	$[-10, 10]^D$	0
F16	Composite F9 + F1 Function V2	$[-100, 100]^D$	0
F17	Composite F9 + F3 Function V2	$[-100, 100]^D$	0
F18	Composite F9 + F4 Function V2	$[-5, 5]^D$	0
F19	Composite F10 + F7 Function V2	$[-10, 10]^D$	0

- **GODE** [36]: A Generalized Opposition-based learning Differential Evolution algorithm. This technique is based on opposition-based learning, which is used to transform candidates from the current search region into new search regions. These transformations are aimed at enabling the algorithm to have a greater chance of finding better solutions than when searching without opposition based transformation.
- **SaDE-MMTS** [43]: A Self-adaptive Differential Evolution algorithm hybridized with a Modified Multi-Trajectory Search strategy (MMTS). This search strategy enhances the search performed by the original SaDE algorithm [28] by frequently refining several diversely distributed solutions at different search stages by using MMTS, satisfying both global and local search requirements.
- **SaEPSDE-MMTS** [42]: An Ensemble of Parameters and mutation Strategies in Differential Evolution with Self-adaption [25] improved with the MMTS, with the aim of enhancing the behavior of the original algorithm.
- **SOUPDE** [37]: Shuffle Or Update Parallel Differential Evolution is a structured population algorithm characterized by sub-populations employing a Differential evolution logic and two strategies: Shuffling, which consists of merging the sub-populations and subsequently randomly dividing them again into sub-populations; and update, which consists of randomly updating the values of the scale factors of each population.
- **GADE** [40]: A Generalized Adaptive Differential Evolution algorithm, which is governed by a generalized parameter adaptation scheme. An auto-adaptive probability distribution, updated during the whole evolution process, is used to generate suitable values for the most important parameters of the underlying Differential Evolution based search procedure.
- **jDElscop** [6]: A self-adaptive Differential Evolution for large scale continuous optimization problems. This is an upgrade of the original jDE algorithm [5], incorporating three different evolution strategies, a population size reduction mechanism, and a mechanism for changing the sign of control parameters.

All methods have been used considering the default configuration provided by their authors in their original submissions to the special issue. Hence, no explicit optimization of parameters was performed.

5. Results and analysis

The experimental study is split into two different sections. The first one (Section 5.1) shows the results obtained after carrying out 25 independent runs of each algorithm over each function. For each run, $5000 \cdot D$ evaluations have been allowed, where D is the number of dimensions of the function (50, 100 or 200).

After performing the analysis based on the ending point of the algorithms, the second section (Section 5.2) performs an analysis of the convergence behavior of the algorithms, using Page's trend test (the original and the alternative version). Differences between the conclusions drawn from the two studies will be pointed out, highlighting the role of the Page's trend test based approach to convergence analysis and the benefits of the alternative version proposed.

5.1. Final results analysis: Friedman and Bergmann tests

Table 5 summarizes the final results obtained by the algorithms in 50, 100 and 200 dimensions' functions, depicted as the number of functions for which the average final error is lower than 1.00E−10 (that is, the number of those for which it can be assumed that the algorithm has reached the optimum). For further reference, full results are included in B).

The final results can be contrasted by using tests for $N \cdot N$ comparisons [12]. In this case, we will use the Friedman test to contrast the differences, and the Bergmann post hoc procedure for adjusting the results for 1×1 pairwise comparisons. Table 6 shows the results of the Friedman test, whereas Table 7 shows the results of the Bergmann post hoc procedure.

As shown by the tables, there are very few differences in the performance of the algorithms. If only the final results are analyzed, only small differences can be found in favor of SOUPDE and jDElscop in every case. However, these differences are not significant.

The p-values computed by the Friedman test shows that there is no significant difference among the algorithms, even at a $\alpha = 0.1$ level of significance. The best (lower) ranks are also obtained by SOUPDE and jDElscop, but this does not make the differences between them and the rest significant. The Bergman procedure supports these conclusions, pointing out that no significant difference can be detected in any pairwise comparison.

Therefore, the conclusions of the study – if only the final results were to be analyzed - would be that all the algorithms exhibit a similar behavior. Perhaps it would be possible to point out that the SOUPDE and jDElscop algorithms show some differences when compared with the rest, but in every case the differences are not significant.

Table 5
Number of functions solved (reached an average error lower than 1.00E−10) per algorithm.

	GODE	SaDE-MMTS	SaEPSDE-MMTS	SOUPDE	GADE	jDElscop
# Functions solved (50-D)	9	11	9	13	8	13
# Functions solved (100-D)	9	9	9	13	8	13
# Functions solved (200-D)	7	6	8	13	7	12

Table 6
Friedman test for the results obtained at 50, 100 and 200 dimensions.

Algorithm	Rank 50-D	Rank 100-D	Rank 200-D
GODE	4.0000	3.9211	4.0263
SaDE-MMTS	3.5789	3.5789	3.9474
SaEPSDE-MMTS	3.5526	3.3684	3.3421
SOUPDE	3.2895	3.3158	3.1842
GADE	3.7632	4.0263	3.8421
jDElscop	2.8158	2.7895	2.6579
p-Value	0.47138	0.35589	0.17045

Table 7
Pairwise hypotheses analyzed by the Bergmann post hoc procedure.

50-D Hypotheses	*p*-Value	100-D Hypotheses	*p*-Value	200-D Hypotheses	*p*-Value
jDElscop vs GODE	0.76586	jDElscop vs GODE	0.62369	jDElscop vs GODE	0.36248
Rest	1.00000	jDElscop vs GADE	0.62369	jDElscop vs GADE	0.36248
–	–	Rest	1.00000	jDElscop vs SaDE-MMTS	0.36248
–	–	–	–	Rest	1.00000

However, these conclusions might not be satisfactory, particularly if the behavior of algorithms over time is analyzed. For example, Fig. 5 shows how important differences can be found between SOUPDE and jDElscop , even when both algorithms reach the same final result. It is not difficult to find examples where one algorithm is converging much faster than another, but, due to the fixed limit on the number of evaluations, this is not detected if only final results are analyzed.

Throughout the rest of the case study, we will show how this difficulty can be overcomed by using the Page's trend test to analyze convergence.

5.2. Convergence analysis

The same experimental conditions have been considered for this second study: Algorithms and functions to study, evaluations limit and so forth. Considering that our framework consists of 19 different functions to optimize, the number of cut-points has been fixed at 10, one after each 10% of fitness function evaluations (see Section 3.1).

Tables 8–10 show a summarized version of the results obtained (an extended version, including ranks at every cut-point, *L* statistics and *p*-values for each pairwise comparison is provided in C). Results are provided for both the original and the alternative version of the test. Each *p*-value of the tables is computed as the probability of rejection of the hypothesis of equality of convergence, in favor of the alternative that the method in the row converges faster than the method in the column. Rejected hypotheses (at a significance level of $\alpha = 0.1$) are highlighted in bold.

The very first fact to note here is the differences between the original and the alternative version of the test: Although in the most clear cases the results seldom change, this is not the case for some of the comparisons, for which very different *p*-values are obtained (particularly in the results for 100 dimensional functions). As shown in Example 2 (which actually reflects the comparison between GODE and SaEPSDE-MMTS in 100 dimensions functions), the interpretation of the results might change dramatically if the alternative version is not considered.

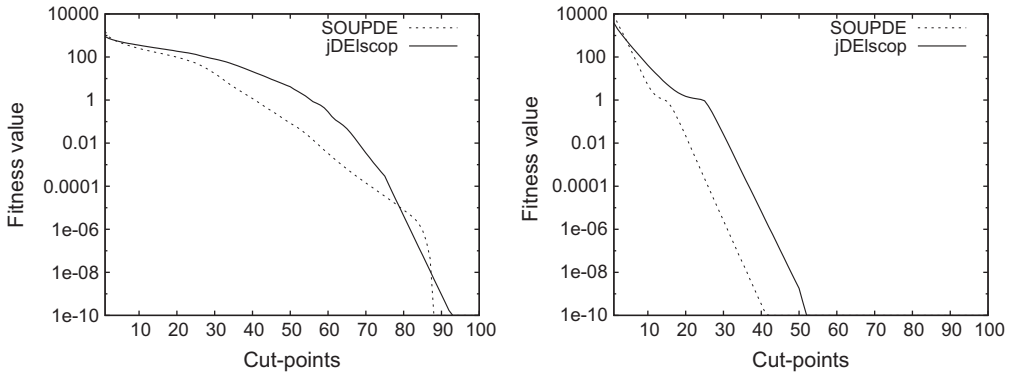

Fig. 5. Convergence plots of SOUPDE vs jDElscop: Performances can be very different even when both algorithms reach the same result at the end. Some algorithms could converge better for most of the run (left graph, 200-*D*, F18), or even finish using 10% less evaluations (right graph, 200-*D*, F5) and traditional final results analysis would not be able to detect these differences in performance.

Table 8
Convergence results (p-values) for the experiments on 50 dimensional functions.

	GODE	SaDE-MMTS	SaEPSDE-MMTS	SOUPDE	GADE	jDElscop
Original version						
GODE	–	**0.00000**	**0.00385**	0.84261	**0.00000**	**0.00000**
SaDE-MMTS	1.00000	–	1.00000	1.00000	1.00000	**0.00000**
SaEPSDE-MMTS	0.99625	**0.00000**	–	0.98330	0.93501	**0.00000**
SOUPDE	0.15940	**0.00000**	**0.01705**	–	**0.00014**	**0.00000**
GADE	1.00000	**0.00000**	**0.06606**	0.99987	–	**0.00000**
jDElscop	1.00000	1.00000	1.00000	1.00000	1.00000	–
Alternative version						
GODE	–	**0.00000**	**0.00997**	0.94811	**0.00000**	**0.00000**
SaDE-MMTS	1.00000	–	0.99999	0.99999	1.00000	**0.00016**
SaEPSDE-MMTS	0.99025	**0.00001**	–	0.90627	0.84758	**0.00000**
SOUPDE	**0.05279**	**0.00001**	**0.09514**	–	**0.00000**	**0.00000**
GADE	1.00000	**0.00000**	0.15439	1.00000	–	**0.00000**
jDElscop	1.00000	0.99985	1.00000	1.00000	1.00000	–

Table 9
Convergence results (p-values) for the experiments on 100 dimensional functions.

	GODE	SaDE-MMTS	SaEPSDE-MMTS	SOUPDE	GADE	jDElscop
Original version						
GODE	–	**0.00451**	**0.05554**	0.83548	**0.00661**	**0.00000**
SaDE-MMTS	0.99560	–	**0.00048**	0.98013	0.41414	**0.00000**
SaEPSDE-MMTS	0.94539	0.99953	–	0.78597	0.76216	**0.00000**
SOUPDE	0.16659	**0.02028**	0.21647	–	**0.06606**	**0.00000**
GADE	0.99354	0.58911	0.24043	0.93501	–	**0.00000**
jDElscop	1.00000	1.00000	1.00000	1.00000	1.00000	–
Alternative version						
GODE	–	0.85907	0.97196	0.90486	0.44361	**0.00000**
SaDE-MMTS	0.14280	–	**0.00004**	0.78108	0.56297	**0.00000**
SaEPSDE-MMTS	**0.02858**	0.99997	–	0.39154	0.97454	**0.00000**
SOUPDE	**0.09656**	0.22139	0.61166	–	**0.01150**	**0.00000**
GADE	0.55968	0.44032	**0.02596**	0.98875	–	**0.00000**
jDElscop	1.00000	1.00000	1.00000	1.00000	1.00000	–

Table 10
Convergence results (p-values) for the experiments on 200 dimensional functions.

	GODE	SaDE-MMTS	SaEPSDE-MMTS	SOUPDE	GADE	jDElscop
Original version						
GODE	–	1.00000	**0.00288**	0.99611	**0.00031**	**0.00000**
SaDE-MMTS	**0.00000**	–	**0.00000**	**0.00000**	**0.00000**	**0.00000**
SaEPSDE-MMTS	0.99719	1.00000	–	0.99753	0.72318	**0.00000**
SOUPDE	**0.00399**	1.00000	**0.00253**	–	**0.00180**	**0.00000**
GADE	0.99970	1.00000	0.27962	0.99825	–	**0.00000**
jDElscop	1.00000	1.00000	1.00000	1.00000	1.00000	–
Alternative version						
GODE	–	0.99999	**0.05889**	0.99995	**0.00046**	**0.00000**
SaDE-MMTS	**0.00001**	–	**0.00003**	0.57771	**0.00004**	**0.00000**
SaEPSDE-MMTS	0.94209	0.99997	–	0.99985	0.82278	**0.00000**
SOUPDE	**0.00006**	0.42555	**0.00016**	–	**0.00000**	**0.00000**
GADE	0.99955	0.99996	0.17940	1.00000	–	**0.00000**
jDElscop	1.00000	1.00000	1.00000	1.00000	1.00000	–

After studying the analysis performed with the alternative version, we can draw the following conclusions:

- 50 dimensions: SOUPDE shows the best convergence behavior, followed by GODE. SaDE-MMTS and jDElscop presents the worst convergence performance in this case.
- 100 dimensions: SOUPDE presents the best convergence in this scenario. GADE and SaDE-MMTS presents a better performance than SaEPSDE-MMTS, and GODE and jDElscop shows the lowest convergence speed in this case.

- 200 dimensions: SaDE-MMTS and SOUPDE are the best methods with respect to convergence in this case. GODE is also significantly better than SaEPSDE-MMTS and GADE, whereas jDElscop presents the worst convergence capabilities.

Although the results differ depending on the number of dimensions considered, it is safe to state that, in general, SOUPDE shows the best behavior with respect to convergence capabilities, whereas jDElscop presents the worst. A further interesting observation is in regard of the relationship between SaDE-MMTS and SaEPSDE-MMTS (the latter is better at 50 dimensions, whereas the former is at 100 dimensions and particularly at 200 dimensions).

The analysis of final results can now be refined with this convergence study. Considering both methodologies, SOUPDE shows the best performance out of the 6 differential evolution methods analyzed. The most striking difference can be found in the results of jDElscop, which has shown a marginal advantage with respect to the final results, but the worst convergence performance. This could indicate that the convergence mechanisms of jDElscop enables it to avoid local optima in a better way than the other methods, but at the cost of a low convergence speed – thus needing more functions evaluations to fully reach its best performance.

Other conclusions include the fact that, despite the similar results obtained by SaEPSDE-MMTS and SaDE-MMTS, the former should be preferred for low dimensional problems whereas SaDE-MMTS should be chosen when the number of dimensions increases. Also, GODE and GADE show a poorer performance than the rest, although the former should still be considered for low dimensional problems.

In summary, these conclusions reveal the fact that useful information about the performance of evolutionary methods in continuous optimization can be drawn if the intermediate results are analyzed. By studying convergence in depth, analyzing how the methods' results evolve as the fitness function evaluations are consumed, new comparisons can be made depicting other useful properties of the search methods rather than just the final results at a predefined point.

Page's trend test has been shown to be a useful method to perform this analysis. Also, the alternative version developed has helped us to mitigate the problem of performing proper comparisons of algorithms that reach the optimum before the maximum fitness evaluation count, enabling us to draw meaningful conclusions about the performances of the methods.

6. Conclusions

In this paper we have presented a new way of analyzing the behavior of EAs in optimization problems, with respect to their convergence performance. We have shown how Page's trend test can be used to perform such an analysis. Also, we have described how the ranks can be computed in an alternative way, in the event that the optimum value of the functions could be reached by the algorithms before the end of the run.

As with other applications of nonparametric tests, the present one does not rely on the assumptions of normality, independence and homoscedasticity. Hence, it is safe to assume that it can be used to analyze the convergence performance of EAs, provided that intermediate results are gathered.

By analyzing such intermediate results, Page's trend test is able to provide key information about algorithms' behavior. Such information can be decisive when establishing differences between algorithms which would otherwise be considered to be equal, if only the final results were used. Therefore, Page's trend test may be regarded as a way of enriching experimental analysis, incorporating statistical convergence analysis within the range of methodologies available.

Acknowledgment

This work was supported by the Spanish Ministry of Education and Science under Grant TIN2011-28488.

Appendix A. Software for statistical convergence analysis

A Java implementation of our approach is available at the SCI2S thematic public website on Statistical Inference in Computational Intelligence and Data Mining, `http://sci2s.ugr.es/sicidm/`. It can be downloaded at the following link: `http://sci2s.ugr.es/sicidm/pageTest.zip`

Its main features are that it:

- Includes both the basic version of Page's trend test and the alternative version.
- Allows us to perform multiple pairwise tests of several algorithms.
- Accepts comma separated values files (CSV) as input data.
- Obtains results as a full report in LaTeX and plain text formats.

The source code is also offered under the terms of the GNU General Public License.

The input data should consist of a CSV file per algorithm, containing the average results obtained at several cutpoints (columns) in several functions (rows). For example, the following file:

```
2.08E+01,1.99E+01,1.86E+01,1.73E+01
3.58E+03,3.38E+03,2.40E+03,9.31E+00
1.11E+00,1.07E+00,1.04E+00,1.02E+00
2.43E−04,2.33E−04,7.85E−05,8.46E−07
0.00E+00,0.00E+00,0.00E+00,0.00E+00
```

represents the results of one algorithm over 5 different functions, taken at 4 different cutpoints (ordered from left to right). All files created should share the same format (number of functions and cutpoints).

Appendix B. Final results of the case of study

The following tables show the final results obtained by each technique in the case of study (for 50, 100 and 200 dimensions). For each pair function/technique, the tables report the average error obtained in 25 independent runs.

Note that if an algorithm reaches an average error lower than 1.00E−10, then it is assumed to have reached the optimum (and thus it is replaced by 0.00E+00). Optimum values in these table are highlighted in bold, and the last row (# Solved) shows the number of optima reached by every algorithm over the 19 functions (see Tables B.11, B.12 and B.13).

Table B.11
Final results at 50 dimensional functions.

	GODE	SaDE-MMTS	SaEPSDE-MMTS	SOUPDE	GADE	jDElscop
F1	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F2	5.02E−02	**0.00E+00**	9.02E−03	1.83E+00	6.43E+00	8.83E−03
F3	3.12E+01	4.43E+00	1.33E+00	2.38E+01	1.85E+01	1.90E+01
F4	**0.00E+00**	**0.00E+00**	4.98E−02	1.19E+00	**0.00E+00**	**0.00E+00**
F5	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F6	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.67E−04	**0.00E+00**
F7	**0.00E+00**	**0.00E+00**	8.07E+01	**0.00E+00**	**0.00E+00**	**0.00E+00**
F8	1.91E−01	7.09E−07	2.30E−06	1.21E−01	9.51E−07	2.50E−02
F9	6.37E−06	1.71E−01	2.03E−01	**0.00E+00**	7.68E−03	**0.00E+00**
F10	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F11	7.15E−06	2.03E−01	1.97E−01	**0.00E+00**	4.99E−03	**0.00E+00**
F12	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F13	2.66E+01	2.54E+01	1.17E+00	2.14E+01	2.08E+01	1.40E+01
F14	4.97E−02	7.23E−03	**0.00E+00**	**0.00E+00**	1.07E−08	**0.00E+00**
F15	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F16	1.66E−10	**0.00E+00**	**0.00E+00**	**0.00E+00**	6.06E−08	**0.00E+00**
F17	1.43E+00	3.94E+00	3.00E−01	1.46E−01	2.90E+00	1.58E−02
F18	4.97E−02	5.15E−02	8.20E−02	**0.00E+00**	1.06E−04	4.97E−02
F19	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
# Solved	9	11	9	13	8	13

Table B.12
Final results at 100 dimensional functions.

	GODE	SaDE-MMTS	SaEPSDE-MMTS	SOUPDE	GADE	jDElscop
F1	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F2	2.18E+00	**0.00E+00**	8.66E−01	8.48E+00	3.03E+01	5.99E−01
F3	8.08E+01	3.91E+01	2.55E+01	7.51E+01	6.80E+01	6.51E+01
F4	**0.00E+00**	**0.00E+00**	**0.00E+00**	4.97E−02	**0.00E+00**	4.97E−02
F5	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F6	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F7	**0.00E+00**	**0.00E+00**	1.62E+02	**0.00E+00**	**0.00E+00**	**0.00E+00**
F8	7.59E+01	4.09E−03	1.22E−01	7.40E+01	6.81E−03	5.94E+00
F9	1.53E−05	2.64E+00	1.05E+00	**0.00E+00**	7.23E−03	**0.00E+00**
F10	**0.00E+00**	1.57E−01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F11	1.45E−05	2.13E+00	1.15E+00	**0.00E+00**	1.16E−02	**0.00E+00**
F12	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	4.51E−10	**0.00E+00**
F13	6.30E+01	4.10E+01	2.47E+01	5.82E+01	2.36E+04	5.14E+01
F14	7.00E−02	1.08E−03	8.88E−03	**0.00E+00**	1.98E−08	**0.00E+00**
F15	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F16	3.63E−10	**0.00E+00**	**0.00E+00**	**0.00E+00**	3.77E−08	**0.00E+00**
F17	1.29E+01	3.71E+00	1.95E+00	7.76E+00	2.43E+01	1.75E−01
F18	1.07E−06	1.67E−01	1.47E−01	**0.00E+00**	7.61E−05	**0.00E+00**
F19	**0.00E+00**	1.05E−01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
# Solved	9	9	9	13	8	13

Table B.13
Final results at 200 dimensional functions.

	GODE	SaDE-MMTS	SaEPSDE-MMTS	SOUPDE	GADE	jDElscop
F1	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F2	1.47E+01	**0.00E+00**	4.58E+00	2.41E+01	4.88E+01	5.56E+00
F3	1.79E+02	4.86E+01	4.39E+01	1.75E+02	1.71E+02	1.42E+02
F4	9.95E−02	5.47E−01	4.98E−02	**0.00E+00**	**0.00E+00**	**0.00E+00**
F5	**0.00E+00**	4.95E−04	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F6	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F7	**0.00E+00**	**0.00E+00**	3.04E+02	**0.00E+00**	**0.00E+00**	**0.00E+00**
F8	2.08E+03	2.72E−01	1.12E+02	2.39E+03	5.14E+00	3.45E+02
F9	3.18E−05	1.27E+01	3.67E+00	**0.00E+00**	1.59E−02	**0.00E+00**
F10	**0.00E+00**	3.33E+00	**0.00E+00**	**0.00E+00**	5.25E−02	**0.00E+00**
F11	3.24E−05	9.95E+00	3.80E+00	1.51E−03	1.17E−02	2.59E−08
F12	1.21E−10	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.11E−09	**0.00E+00**
F13	1.39E+02	5.49E+01	1.03E+02	1.34E+02	1.28E+02	1.21E+02
F14	5.99E−02	5.71E−01	4.96E−02	**0.00E+00**	6.29E−08	5.36E−04
F15	**0.00E+00**	4.41E−01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F16	7.88E−10	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.74E−08	**0.00E+00**
F17	3.84E+01	1.62E+01	2.33E+01	3.33E+01	3.90E+01	2.07E+01
F18	8.01E−02	1.41E+00	5.58E−01	**0.00E+00**	8.90E−05	**0.00E+00**
F19	**0.00E+00**	2.78E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
# Solved	7	6	8	13	7	12

Appendix C. Full results of the Page's trend test

The following tables shows the full results obtained in each application of Page's trend test for the case study. For each pairwise comparison, the average ranks at 10 cutpoints, the L statistic and the p-value computed are reported.

The results include all the possible pairwise comparisons at 50, 100 and 200 dimensions. Both versions of the test (the original and the alternative) are considered (see Tables C.14, C.15, C.16, C.17, C.18 and C.19).

Table C.14
Full results of Page's trend test on 50 dimensional functions.

Algorithms	L statistic	p-Value	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
GODE vs SaDE-MMTS	6353.0	0.00000	70.0	80.0	86.0	94.0	98.5	110.5	116.0	124.0	130.5	135.5
GODE vs SaEPSDE-MMTS	6067.5	0.00385	73.0	88.0	86.0	102.0	116.5	117.5	135.0	121.0	108.0	98.0
GODE vs SOUPDE	5627.5	0.84261	80.0	112.0	126.0	119.0	115.0	112.0	99.5	97.5	94.0	90.0
GODE vs GADE	6437.5	0.00000	58.0	75.0	98.0	100.0	88.5	106.5	117.0	125.0	135.0	142.0
GODE vs jDElscop	7032.5	0.00000	21.0	51.0	70.0	84.0	94.0	117.0	137.0	159.0	159.5	152.5
SaDE-MMTS vs GODE	5142.0	1.00000	139.0	129.0	123.0	115.0	110.5	98.5	93.0	85.0	78.5	73.5
SaDE-MMTS vs SaEPSDE-MMTS	5201.0	1.00000	111.0	133.0	132.0	125.0	113.0	93.0	104.5	91.5	78.5	63.5
SaDE-MMTS vs SOUPDE	5111.5	1.00000	140.0	129.0	123.0	120.0	110.5	96.5	95.0	83.0	76.0	72.0
SaDE-MMTS vs GADE	5040.0	1.00000	133.0	132.0	137.0	127.0	110.5	98.5	86.5	78.5	73.0	69.0
SaDE-MMTS vs jDElscop	6419.5	0.00000	28.0	42.0	103.0	138.0	128.0	128.0	125.0	124.0	118.5	110.5
SaEPSDE-MMTS vs GODE	5427.5	0.99625	136.0	121.0	123.0	107.0	92.5	91.5	74.0	88.0	101.0	111.0
SaEPSDE-MMTS vs SaDE-MMTS	6294.0	0.00000	98.0	76.0	77.0	84.0	96.0	116.0	104.5	117.5	130.5	145.5
SaEPSDE-MMTS vs SOUPDE	5493.0	0.98330	127.0	121.0	118.0	106.0	95.5	95.5	80.5	90.5	102.0	109.0
SaEPSDE-MMTS vs GADE	5566.5	0.93501	127.0	122.0	117.0	106.0	86.0	87.0	77.5	95.5	108.0	119.0
SaEPSDE-MMTS vs jDElscop	7012.5	0.00000	30.0	36.0	55.0	95.0	112.0	140.0	123.0	141.0	153.5	159.5
SOUPDE vs GODE	5867.5	0.15940	129.0	97.0	83.0	90.0	94.0	97.0	109.5	111.5	115.0	119.0
SOUPDE vs SaDE-MMTS	6383.5	0.00000	69.0	80.0	86.0	89.0	98.5	112.5	114.0	126.0	133.0	137.0
SOUPDE vs SaEPSDE-MMTS	6002.0	0.01705	82.0	88.0	91.0	103.0	113.5	113.5	128.5	118.5	107.0	100.0
SOUPDE vs GADE	6184.0	0.00014	99.0	85.0	84.0	97.0	82.5	101.5	112.0	121.0	128.5	134.5
SOUPDE vs jDElscop	7005.5	0.00000	40.0	53.0	62.0	75.0	88.0	112.0	138.0	156.0	162.5	158.5
GADE vs GODE	5057.5	1.00000	151.0	134.0	111.0	109.0	120.5	102.5	92.0	84.0	74.0	67.0
GADE vs SaDE-MMTS	6455.0	0.00000	76.0	77.0	72.0	82.0	98.5	110.5	122.5	130.5	136.0	140.0
GADE vs SaEPSDE-MMTS	5928.5	0.06606	82.0	87.0	92.0	103.0	123.0	122.0	131.5	113.5	101.0	90.0
GADE vs SOUPDE	5311.0	0.99987	110.0	124.0	125.0	112.0	126.5	107.5	97.0	88.0	80.5	74.5
GADE vs jDElscop	6905.0	0.00000	32.0	42.0	61.0	82.0	108.0	147.0	144.5	148.5	143.5	136.5
jDElscop vs GODE	4462.5	1.00000	188.0	158.0	139.0	125.0	115.0	92.0	72.0	50.0	49.5	56.5
jDElscop vs SaDE-MMTS	5075.5	1.00000	181.0	167.0	106.0	71.0	81.0	81.0	84.0	85.0	90.5	98.5
jDElscop vs SaEPSDE-MMTS	4482.5	1.00000	179.0	173.0	154.0	114.0	97.0	69.0	86.0	68.0	55.5	49.5
jDElscop vs SOUPDE	4489.5	1.00000	169.0	156.0	147.0	134.0	121.0	97.0	71.0	53.0	46.5	50.5
jDElscop vs GADE	4590.0	1.00000	177.0	167.0	148.0	127.0	101.0	62.0	64.5	60.5	65.5	72.5

Table C.15
Full results of Page's trend test on 50 dimensional functions (alternative version).

Algorithms	*L* statistic	*p*-Value	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
GODE vs SaDE-MMTS	6283.5	0.00000	80.0	83.0	89.0	96.0	95.5	100.5	109.5	122.5	130.5	138.5
GODE vs SaEPSDE-MMTS	6027.0	0.00997	87.0	95.0	90.0	93.0	103.0	107.0	129.5	123.5	113.5	103.5
GODE vs SOUPDE	5553.0	0.94811	93.0	116.0	123.0	116.0	112.0	106.0	99.5	99.5	93.5	86.5
GODE vs GADE	6547.5	0.00000	59.0	75.0	83.0	90.0	95.0	106.0	118.5	127.5	141.0	150.0
GODE vs jDElscop	6547.0	0.00000	41.0	64.0	83.0	97.0	110.0	127.0	140.5	124.5	130.5	127.5
SaDE-MMTS vs GODE	5211.5	1.00000	129.0	126.0	120.0	113.0	113.5	108.5	99.5	86.5	78.5	70.5
SaDE-MMTS vs SaEPSDE-MMTS	5244.5	0.99999	120.0	135.0	127.0	115.0	106.5	91.5	98.5	92.5	87.5	71.5
SaDE-MMTS vs SOUPDE	5229.5	0.99999	130.0	126.0	120.0	115.0	110.5	104.5	96.5	85.5	81.5	75.5
SaDE-MMTS vs GADE	5006.0	1.00000	130.0	136.0	137.0	128.0	114.0	101.0	86.0	78.0	71.0	64.0
SaDE-MMTS vs jDElscop	6179.5	0.00016	51.0	62.0	103.0	126.0	126.0	130.0	118.0	112.0	110.5	106.5
SaEPSDE-MMTS vs GODE	5468.0	0.99025	122.0	114.0	119.0	116.0	106.0	102.0	79.5	85.5	95.5	105.5
SaEPSDE-MMTS vs SaDE-MMTS	6250.5	0.00001	89.0	74.0	82.0	94.0	102.5	117.5	110.5	116.5	121.5	137.5
SaEPSDE-MMTS vs SOUPDE	5590.0	0.90627	114.0	112.0	112.0	113.0	102.0	101.0	90.0	96.0	100.0	105.0
SaEPSDE-MMTS vs GADE	5625.0	0.84758	114.0	116.0	111.0	109.0	103.0	100.0	78.5	92.5	104.5	116.5
SaEPSDE-MMTS vs jDElscop	6655.0	0.00000	44.0	50.0	69.0	103.0	120.0	140.0	119.0	124.0	133.0	143.0
SOUPDE vs GODE	5942.0	0.05279	116.0	93.0	86.0	93.0	97.0	103.0	109.5	109.5	115.5	122.5
SOUPDE vs SaDE-MMTS	6265.5	0.00001	79.0	83.0	89.0	94.0	98.5	104.5	112.5	123.5	127.5	133.5
SOUPDE vs SaEPSDE-MMTS	5905.0	0.09514	95.0	97.0	97.0	96.0	107.0	108.0	119.0	113.0	109.0	104.0
SOUPDE vs GADE	6471.0	0.00000	82.0	75.0	74.0	86.0	91.0	103.0	114.0	129.0	140.0	151.0
SOUPDE vs jDElscop	6925.0	0.00000	43.0	56.0	65.0	78.0	94.0	113.0	134.5	142.5	156.5	162.5
GADE vs GODE	4947.5	1.00000	150.0	134.0	126.0	119.0	114.0	103.0	90.5	81.5	68.0	59.0
GADE vs SaDE-MMTS	6489.0	0.00000	79.0	73.0	72.0	81.0	95.0	108.0	123.0	131.0	138.0	145.0
GADE vs SaEPSDE-MMTS	5870.0	0.15439	95.0	93.0	98.0	100.0	106.0	109.0	130.5	116.5	104.5	92.5
GADE vs SOUPDE	5024.0	1.00000	127.0	134.0	135.0	123.0	118.0	106.0	95.0	80.0	69.0	58.0
GADE vs jDElscop	6469.0	0.00000	48.0	58.0	77.0	98.0	118.0	144.0	141.0	124.0	121.0	116.0
jDElscop vs GODE	4948.0	1.00000	168.0	145.0	126.0	112.0	99.0	82.0	68.5	84.5	78.5	81.5
jDElscop vs SaDE-MMTS	5315.5	0.99985	158.0	147.0	106.0	83.0	83.0	79.0	91.0	97.0	98.5	102.5
jDElscop vs SaEPSDE-MMTS	4840.0	1.00000	165.0	159.0	140.0	106.0	89.0	69.0	90.0	85.0	76.0	66.0
jDElscop vs SOUPDE	4570.0	1.00000	166.0	153.0	144.0	131.0	115.0	96.0	74.5	66.5	52.5	46.5
jDElscop vs GADE	5026.0	1.00000	161.0	151.0	132.0	111.0	91.0	65.0	68.0	85.0	88.0	93.0

Table C.16
Full results of Page's trend test on 100 dimensional functions.

Algorithms	*L* statistic	*p*-Value	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
GODE vs SaDE-MMTS	6061.0	0.00451	82.0	81.0	85.0	110.0	127.0	114.0	105.5	113.5	113.5	113.5
GODE vs SaEPSDE-MMTS	5939.0	0.05554	79.0	93.0	93.0	108.0	115.0	114.0	129.5	116.5	104.5	92.5
GODE vs SOUPDE	5631.0	0.83548	72.0	121.0	124.0	127.0	114.0	104.0	98.0	99.0	95.0	91.0
GODE vs GADE	6045.0	0.00661	76.0	91.0	100.0	109.0	109.0	106.0	108.5	115.5	113.5	116.5
GODE vs jDElscop	7029.0	0.00000	23.0	53.0	68.0	79.0	93.0	119.0	140.5	160.5	156.5	152.5
SaDE-MMTS vs GODE	5434.0	0.99560	127.0	128.0	124.0	99.0	82.0	95.0	103.5	95.5	95.5	95.5
SaDE-MMTS vs SaEPSDE-MMTS	6143.5	0.00048	46.0	99.0	105.5	103.5	108.0	112.0	132.5	121.5	112.5	104.5
SaDE-MMTS vs SOUPDE	5501.5	0.98013	100.0	124.0	127.0	109.0	102.5	102.5	101.0	98.0	92.0	89.0
SaDE-MMTS vs GADE	5774.0	0.41414	91.0	95.0	122.0	115.0	103.0	104.0	104.5	102.5	103.5	104.5
SaDE-MMTS vs jDElscop	6944.5	0.00000	28.0	37.0	66.0	94.0	104.0	131.0	150.5	146.5	143.0	145.0
SaEPSDE-MMTS vs GODE	5556.0	0.94539	130.0	116.0	116.0	101.0	94.0	95.0	79.5	92.5	104.5	116.5
SaEPSDE-MMTS vs SaDE-MMTS	5351.5	0.99953	163.0	110.0	103.5	105.5	101.0	97.0	76.5	87.5	96.5	104.5
SaEPSDE-MMTS vs SOUPDE	5653.0	0.78597	125.0	107.0	112.0	98.0	98.0	97.0	87.0	98.0	109.0	114.0
SaEPSDE-MMTS vs GADE	5662.5	0.76216	118.0	123.0	108.0	95.0	92.5	96.5	86.5	99.5	108.5	117.5
SaEPSDE-MMTS vs jDElscop	7121.0	0.00000	21.0	41.0	55.0	84.0	103.0	141.0	131.5	145.5	158.5	164.5
SOUPDE vs GODE	5864.0	0.16659	137.0	88.0	85.0	82.0	95.0	105.0	111.0	110.0	114.0	118.0
SOUPDE vs SaDE-MMTS	5993.5	0.02028	109.0	85.0	82.0	100.0	106.5	106.5	108.0	111.0	117.0	120.0
SOUPDE vs SaEPSDE-MMTS	5842.0	0.21647	84.0	102.0	97.0	111.0	111.0	112.0	122.0	111.0	100.0	95.0
SOUPDE vs GADE	5928.5	0.06606	106.0	95.0	95.0	98.0	98.5	101.5	107.5	109.5	114.5	119.5
SOUPDE vs jDElscop	6858.0	0.00000	42.0	60.0	70.0	87.0	89.0	110.0	130.0	146.0	155.0	156.0
GADE vs GODE	5450.0	0.99354	133.0	118.0	109.0	100.0	100.0	103.0	100.5	93.5	95.5	92.5
GADE vs SaDE-MMTS	5721.0	0.58911	118.0	114.0	87.0	94.0	106.0	105.0	104.5	106.5	105.5	104.5
GADE vs SaEPSDE-MMTS	5832.5	0.24043	91.0	86.0	101.0	114.0	116.5	112.5	122.5	109.5	100.5	91.5
GADE vs SOUPDE	5566.5	0.93501	103.0	114.0	114.0	111.0	110.5	107.5	101.5	99.5	94.5	89.5
GADE vs jDElscop	6920.0	0.00000	32.0	42.0	61.0	79.0	113.0	145.0	142.5	144.5	143.5	142.5
jDElscop vs GODE	4466.0	1.00000	186.0	156.0	141.0	130.0	116.0	90.0	68.5	48.5	52.5	56.5
jDElscop vs SaDE-MMTS	4550.5	1.00000	181.0	172.0	143.0	115.0	105.0	78.0	58.5	62.5	66.0	64.0
jDElscop vs SaEPSDE-MMTS	4374.0	1.00000	188.0	168.0	154.0	125.0	106.0	68.0	77.5	63.5	50.5	44.5
jDElscop vs SOUPDE	4637.0	1.00000	167.0	149.0	139.0	122.0	120.0	99.0	79.0	63.0	54.0	53.0
jDElscop vs GADE	4575.0	1.00000	177.0	167.0	148.0	130.0	96.0	64.0	66.5	64.5	65.5	66.5

Table C.17
Full results of Page's trend test on 100 dimensional functions (alternative version).

Algorithms	*L* statistic	*p*-Value	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
GODE vs SaDE-MMTS	5619.0	0.85907	111.0	110.0	108.0	107.0	109.0	102.0	99.5	101.5	99.5	97.5
GODE vs SaEPSDE-MMTS	5519.0	0.97196	111.0	118.0	112.0	102.0	100.0	105.0	122.5	106.5	91.5	76.5
GODE vs SOUPDE	5591.0	0.90486	85.0	119.0	122.0	125.0	112.0	102.0	95.0	94.0	97.0	94.0
GODE vs GADE	5765.0	0.44361	99.0	106.0	104.0	103.0	108.0	107.0	105.5	106.5	102.5	103.5
GODE vs jDElscop	6535.0	0.00000	41.0	66.0	81.0	95.0	109.0	131.0	144.5	126.5	125.5	125.5
SaDE-MMTS vs GODE	5876.0	0.14280	98.0	99.0	101.0	102.0	100.0	107.0	109.5	107.5	109.5	111.5
SaDE-MMTS vs SaEPSDE-MMTS	6224.5	0.00004	55.0	94.0	97.0	94.0	104.5	111.5	132.5	125.5	118.5	112.5
SaDE-MMTS vs SOUPDE	5655.0	0.78108	95.0	109.0	113.0	114.0	114.0	107.0	100.5	98.5	96.5	97.5
SaDE-MMTS vs GADE	5729.0	0.56297	96.0	100.0	116.0	109.0	110.0	107.0	104.0	100.0	101.0	102.0
SaDE-MMTS vs jDElscop	6421.0	0.00000	51.0	60.0	81.0	109.0	117.0	130.0	136.5	118.5	117.5	124.5
SaEPSDE-MMTS vs GODE	5976.0	0.02858	98.0	91.0	97.0	107.0	109.0	104.0	86.5	102.5	117.5	132.5
SaEPSDE-MMTS vs SaDE-MMTS	5270.5	0.99997	154.0	115.0	112.0	115.0	104.5	97.5	76.5	83.5	90.5	96.5
SaEPSDE-MMTS vs SOUPDE	5781.0	0.39154	103.0	95.0	103.0	110.0	112.0	109.0	101.0	99.0	104.0	109.0
SaEPSDE-MMTS vs GADE	5514.0	0.97454	122.0	127.0	112.0	99.0	99.0	104.0	83.5	91.5	99.5	107.5
SaEPSDE-MMTS vs jDElscop	6639.0	0.00000	37.0	57.0	71.0	100.0	119.0	137.0	133.5	126.5	128.5	135.5
SOUPDE vs GODE	5904.0	0.09656	124.0	90.0	87.0	84.0	97.0	107.0	114.0	115.0	112.0	115.0
SOUPDE vs SaDE-MMTS	5840.0	0.22139	114.0	100.0	96.0	95.0	95.0	102.0	108.5	110.5	112.5	111.5
SOUPDE vs SaEPSDE-MMTS	5714.0	0.61166	106.0	114.0	106.0	99.0	97.0	100.0	108.0	110.0	105.0	100.0
SOUPDE vs GADE	6020.5	0.01150	98.0	92.0	92.0	95.0	100.5	105.5	109.5	113.5	117.5	121.5
SOUPDE vs jDElscop	6794.5	0.00000	48.0	66.0	76.0	87.0	86.0	101.0	124.0	140.0	155.5	161.5
GADE vs GODE	5730.0	0.55968	110.0	103.0	105.0	106.0	101.0	102.0	103.5	102.5	106.5	105.5
GADE vs SaDE-MMTS	5766.0	0.44032	113.0	109.0	93.0	100.0	99.0	102.0	105.0	109.0	108.0	107.0
GADE vs SaEPSDE-MMTS	5981.0	0.02596	87.0	82.0	97.0	110.0	110.0	105.0	125.5	117.5	109.5	101.5
GADE vs SOUPDE	5474.5	0.98875	111.0	117.0	117.0	114.0	108.5	103.5	99.5	95.5	91.5	87.5
GADE vs jDElscop	6330.0	0.00000	55.0	65.0	84.0	102.0	124.0	135.0	132.5	116.5	114.5	116.5
jDElscop vs GODE	4960.0	1.00000	168.0	143.0	128.0	114.0	100.0	78.0	64.5	82.5	83.5	83.5
jDElscop vs SaDE-MMTS	5074.0	1.00000	158.0	149.0	128.0	100.0	92.0	79.0	72.5	90.5	91.5	84.5
jDElscop vs SaEPSDE-MMTS	4856.0	1.00000	172.0	152.0	138.0	109.0	90.0	72.0	75.5	82.5	80.5	73.5
jDElscop vs SOUPDE	4700.5	1.00000	161.0	143.0	133.0	122.0	123.0	108.0	85.0	69.0	53.5	47.5
jDElscop vs GADE	5165.0	1.00000	154.0	144.0	125.0	107.0	85.0	74.0	76.5	92.5	94.5	92.5

Table C.18
Full results of Page's trend test on 200 dimensional functions.

Algorithms	*L* statistic	*p*-value	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
GODE vs SaDE-MMTS	4753.0	1.00000	190.0	144.0	119.0	116.0	98.5	89.5	78.0	76.0	69.5	64.5
GODE vs SaEPSDE-MMTS	6079.0	0.00288	71.0	90.0	98.0	104.0	109.5	114.5	117.5	115.5	113.0	112.0
GODE vs SOUPDE	5429.0	0.99611	89.0	124.0	139.0	128.0	109.5	100.5	90.0	90.0	87.5	87.5
GODE vs GADE	6158.0	0.00031	76.0	83.0	94.0	111.0	103.5	105.5	110.5	114.5	120.0	127.0
GODE vs jDElscop	7009.5	0.00000	22.0	50.0	70.0	84.0	94.0	123.0	139.5	156.5	155.0	151.0
SaDE-MMTS vs GODE	6742.0	0.00000	19.0	65.0	90.0	93.0	110.5	119.5	131.0	133.0	139.5	144.5
SaDE-MMTS vs SaEPSDE-MMTS	6541.0	0.00000	28.0	69.0	90.5	107.5	122.0	115.0	126.0	125.0	128.5	133.5
SaDE-MMTS vs SOUPDE	6479.0	0.00000	19.0	90.0	101.0	99.0	113.5	112.5	127.5	125.5	128.0	129.0
SaDE-MMTS vs GADE	6650.0	0.00000	19.0	67.0	90.0	112.0	113.0	114.0	126.5	128.5	133.5	141.5
SaDE-MMTS vs jDElscop	7129.5	0.00000	19.0	38.0	57.0	86.0	106.0	130.0	142.5	147.5	158.0	161.0
SaEPSDE-MMTS vs GODE	5416.0	0.99719	138.0	119.0	111.0	105.0	99.5	94.5	91.5	93.5	96.0	97.0
SaEPSDE-MMTS vs SaDE-MMTS	4954.0	1.00000	181.0	140.0	118.5	101.5	87.0	94.0	83.0	84.0	80.5	75.5
SaEPSDE-MMTS vs SOUPDE	5411.0	0.99753	132.0	119.0	118.0	105.0	100.5	93.5	93.5	95.5	95.0	93.0
SaEPSDE-MMTS vs GADE	5677.0	0.72318	111.0	107.0	106.0	114.0	102.0	97.0	96.0	101.0	104.0	107.0
SaEPSDE-MMTS vs jDElscop	6908.0	0.00000	34.0	49.0	62.0	84.0	102.0	138.0	136.0	144.0	148.0	148.0
SOUPDE vs GODE	6066.0	0.00399	120.0	85.0	70.0	81.0	99.5	108.5	119.0	119.0	121.5	121.5
SOUPDE vs SaDE-MMTS	5016.0	1.00000	190.0	119.0	108.0	110.0	95.5	96.5	81.5	83.5	81.0	80.0
SOUPDE vs SaEPSDE-MMTS	6084.0	0.00253	77.0	90.0	91.0	104.0	108.5	115.5	115.5	113.5	114.0	116.0
SOUPDE vs GADE	6097.0	0.00180	106.0	87.0	82.0	95.0	92.5	102.5	109.5	117.5	123.0	130.0
SOUPDE vs jDElscop	6952.5	0.00000	27.0	53.0	76.0	86.0	91.0	116.0	134.5	152.5	155.0	154.0
GADE vs GODE	5337.0	0.99970	133.0	126.0	115.0	98.0	105.5	103.5	98.5	94.5	89.0	82.0
GADE vs SaDE-MMTS	4845.0	1.00000	190.0	142.0	119.0	97.0	96.0	95.0	82.5	80.5	75.5	67.5
GADE vs SaEPSDE-MMTS	5818.0	0.27962	98.0	102.0	103.0	95.0	107.0	112.0	113.0	108.0	105.0	102.0
GADE vs SOUPDE	5398.0	0.99825	103.0	122.0	127.0	114.0	116.5	106.5	99.5	91.5	86.0	79.0
GADE vs jDElscop	7018.5	0.00000	31.0	42.0	55.0	74.0	106.0	143.0	147.0	147.0	150.5	149.5
jDElscop vs GODE	4485.5	1.00000	187.0	159.0	139.0	125.0	115.0	86.0	69.5	52.5	54.0	58.0
jDElscop vs SaDE-MMTS	4365.5	1.00000	190.0	171.0	152.0	123.0	103.0	79.0	66.5	61.5	51.0	48.0
jDElscop vs SaEPSDE-MMTS	4587.0	1.00000	175.0	160.0	147.0	125.0	107.0	71.0	73.0	65.0	61.0	61.0
jDElscop vs SOUPDE	4542.5	1.00000	182.0	156.0	133.0	123.0	118.0	93.0	74.5	56.5	54.0	55.0
jDElscop vs GADE	4476.5	1.00000	178.0	167.0	154.0	135.0	103.0	66.0	62.0	62.0	58.5	59.5

Table C.19
Full results of Page's trend test on 200 dimensional functions (alternative version).

Algorithms	L statistic	p-Value	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
GODE vs SaDE-MMTS	5234.5	0.99999	165.0	126.0	101.0	97.0	96.0	98.0	95.0	94.0	88.5	84.5
GODE vs SaEPSDE-MMTS	5935.5	0.05889	88.0	100.0	105.0	98.0	99.0	107.0	113.5	112.5	111.0	111.0
GODE vs SOUPDE	5284.5	0.99995	105.0	127.0	139.0	128.0	109.5	98.5	89.5	85.5	82.5	80.5
GODE vs GADE	6145.0	0.00046	85.0	83.0	85.0	105.0	107.0	108.0	112.0	114.0	120.0	126.0
GODE vs jDElscop	6617.0	0.00000	37.0	62.0	82.0	96.0	106.0	125.0	140.5	135.5	131.5	129.5
SaDE-MMTS vs GODE	6260.5	0.00001	44.0	83.0	108.0	112.0	113.0	111.0	114.0	115.0	120.5	124.5
SaDE-MMTS vs SaEPSDE-MMTS	6227.5	0.00003	46.0	87.0	96.0	119.0	124.0	106.0	113.0	113.0	117.5	123.5
SaDE-MMTS vs SOUPDE	5724.5	0.57771	57.0	115.0	126.0	125.0	118.0	111.0	114.0	96.0	92.5	90.5
SaDE-MMTS vs GADE	6220.0	0.00004	43.0	83.0	107.0	121.0	117.0	110.0	113.0	112.0	116.0	123.0
SaDE-MMTS vs jDElscop	6284.0	0.00000	50.0	69.0	88.0	117.0	122.0	126.0	128.0	117.0	114.0	114.0
SaEPSDE-MMTS vs GODE	5559.5	0.94209	121.0	109.0	104.0	111.0	110.0	102.0	95.5	96.5	98.0	98.0
SaEPSDE-MMTS vs SaDE-MMTS	5267.5	0.99997	163.0	122.0	113.0	90.0	85.0	103.0	96.0	96.0	91.5	85.5
SaEPSDE-MMTS vs SOUPDE	5315.0	0.99985	121.0	118.0	117.0	117.0	117.0	107.0	105.0	86.0	81.0	76.0
SaEPSDE-MMTS vs GADE	5637.0	0.82278	107.0	110.0	109.0	107.0	112.0	106.0	95.5	97.5	99.5	101.5
SaEPSDE-MMTS vs jDElscop	6507.5	0.00000	47.0	62.0	75.0	97.0	115.0	138.0	136.5	128.5	123.0	123.0
SOUPDE vs GODE	6210.5	0.00006	104.0	82.0	70.0	81.0	99.5	110.5	119.5	123.5	126.5	128.5
SOUPDE vs SaDE-MMTS	5770.5	0.42555	152.0	94.0	83.0	84.0	91.0	98.0	95.0	113.0	116.5	118.5
SOUPDE vs SaEPSDE-MMTS	6180.0	0.00016	88.0	91.0	92.0	92.0	92.0	102.0	104.0	123.0	128.0	133.0
SOUPDE vs GADE	6283.0	0.00000	91.0	84.0	79.0	87.0	94.0	104.0	111.5	124.5	131.5	138.5
SOUPDE vs jDElscop	7032.5	0.00000	27.0	50.0	73.0	86.0	91.0	110.0	130.0	151.0	160.5	166.5
GADE vs GODE	5350.0	0.99955	124.0	126.0	124.0	104.0	102.0	101.0	97.0	95.0	89.0	83.0
GADE vs SaDE-MMTS	5275.0	0.99996	166.0	126.0	102.0	88.0	92.0	99.0	96.0	97.0	93.0	86.0
GADE vs SaEPSDE-MMTS	5858.0	0.17940	102.0	99.0	100.0	102.0	97.0	103.0	113.5	111.5	109.5	107.5
GADE vs SOUPDE	5212.0	1.00000	118.0	125.0	130.0	122.0	115.0	105.0	97.5	84.5	77.5	70.5
GADE vs jDElscop	6438.0	0.00000	54.0	65.0	78.0	97.0	117.0	131.0	132.0	125.0	123.0	123.0
jDElscop vs GODE	4878.0	1.00000	172.0	147.0	127.0	113.0	103.0	84.0	68.5	73.5	77.5	79.5
jDElscop vs SaDE-MMTS	5211.0	1.00000	159.0	140.0	121.0	92.0	87.0	83.0	81.0	92.0	95.0	95.0
jDElscop vs SaEPSDE-MMTS	4987.5	1.00000	162.0	147.0	134.0	112.0	94.0	71.0	72.5	80.5	86.0	86.0
jDElscop vs SOUPDE	4462.5	1.00000	182.0	159.0	136.0	123.0	118.0	99.0	79.0	58.0	48.5	42.5
jDElscop vs GADE	5057.0	1.00000	155.0	144.0	131.0	112.0	92.0	78.0	77.0	84.0	86.0	86.0

References

[1] M.G. Arenas, N. Rico, A.M. Mora, P.A. Castillo, J.J. Merelo, Using statistical tools to determine the significance and relative importance of the main parameters of an evolutionary algorithm, Intell. Data Anal. 17 (2013) 771–789.
[2] T. Bartz-Beielstein, Experimental Research in Evolutionary Computation: The New Experimentalism, Springer, New York, 2006.
[3] B. Bischl, O. Mersmann, H. Trautmann, M. Preuss, Algorithm selection based on exploratory landscape analysis and cost-sensitive learning, in: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, GECCO '12, New York, USA, July 7–11.
[4] P.A.N. Bosman, On gradients and hybrid evolutionary algorithms for real-valued multiobjective optimization, IEEE Trans. Evol. Comput. 16 (2012) 51–69.
[5] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Trans. Evol. Comput. 10 (2006) 646–657.
[6] J. Brest, M.S. Maucec, Self-adaptive differential evolution algorithm using population size reduction and three strategies, Soft Comput. 15 (2011) 2157–2174.
[7] P. Chakraborty, S. Das, G.G. Roy, A. Abraham, On convergence of the multi-objective particle swarm optimizers, Inf. Sci. 181 (2011) 1411–1425.
[8] W.J. Conover, Practical Nonparametric Statistic, third ed., John Wiley & Sons, 1999.
[9] M. Crepinsek, S.H. Liu, M. Mernik, Replication and comparison of computational experiments in applied evolutionary computing: common pitfalls and guidelines to avoid them, Appl. Soft Comput. 19 (2014) 161–170.
[10] A. Czarn, C. MacNish, K. Vijayan, R. Turlach, R. Gupta, Statistical exploratory analysis of genetic algorithms, IEEE Trans. Evol. Comput. 8 (2004) 405–421.
[11] S. Das, P.N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (2011) 4–31.
[12] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (2011) 3–18.
[13] E.A. Duéñez-Guzmán, M.D. Vose, No free lunch and benchmarks, Evolut. Comput. 21 (2013) 293–312.
[14] M. Gallagher, B. Yuan, A general-purpose tunable landscape generator, IEEE Trans. Evol. Comput. 10 (2006) 590–603.
[15] S. García, A. Fernández, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability, Soft Comput. 13 (2009) 959–977.
[16] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power, Inform. Sci. 180 (2010) 2044–2064.
[17] S. García, F. Herrera, An extension on Statistical Comparisons of Classifiers over Multiple Data Sets for all pairwise comparisons, J. Mach. Learn. Res. 9 (2008) 2677–2694.
[18] 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 (2009) 617–644.
[19] J. Gibbons, S. Chakraborti, Nonparametric Statistical Inference, fifth ed., Chapman & Hall, 2010.
[20] F. Herrera, M. Lozano, D. Molina, Test Suite for the Special Issue of Soft Computing on Scalability of Evolutionary Algorithms and Other Metaheuristics for Large Scale Continuous Optimization Problems, 2010. <http://sci2s.ugr.es/eamhco/cfp.php>.
[21] F. Herrera, M. Lozano, D. Molina, Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems, Soft Comput. 15 (2011) 2085–2087.
[22] J.J. Higgins, Introduction to Modern Nonparametric Statistics, Duxbury Press, 2003.
[23] J. Hooker, Testing heuristics: we have it all wrong, J. Heuristics 1 (1997) 33–42.

[24] J. Luengo, S. García, F. Herrera, A study on the use of statistical tests for experimentation with neural networks: analysis of parametric test conditions and non-parametric tests, Expert Syst. Appl. 36 (2009) 7798–7808.
[25] R. Mallipeddi, P.N. Suganthan, Q.K. Pan, M.F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Appl. Soft Comput. 11 (2011) 1679–1696.
[26] O. Mersmann, B. Bischl, H. Trautmann, M. Preuss, C. Weihs, G. Rudolph, Exploratory landscape analysis, in: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, GECCO '11, New York, USA, July 12–16.
[27] E.B. Page, Ordered hypotheses for multiple treatments: a significance test for linear ranks, J. Am. Stat. Assoc. 58 (1963) 216–230.
[28] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2009) 398–417.
[29] I. Rojas, J. González, H. Pomares, J.J. Merelo, P.A. Castillo, G. Romero, Statistical analysis of the main parameters involved in the design of a genetic algorithm, IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev. 32 (2002) 31–37.
[30] G. Rudolph, Convergence analysis of canonical genetic algorithms, IEEE Trans. Neural Networks 5 (1994) 96–101.
[31] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, fifth ed., Chapman & Hall/CRC, 2011.
[32] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. Arch. 11 (1997) 341–359.
[33] P. Suganthan, N. Hansen, J. Liang, K. Deb, Y. Chen, A. Auger, S. Tiwari, Problem Definitions and Evaluation Criteria for the CEC'2005 Special Session on Real Parameter Optimization. Nanyang Technological University, Technical Report, 2005. <www.ntu.edu.sg/home/epnsugan/index_files/cec-05/Tech-Report-May-30-05.pdf>.
[34] K. Tang, X. Yao, P.N. Suganthan, C. MacNish, Y.P. Chen, C.M. Chen, Z. Yang, Benchmark Functions for the CEC'2008 Special Session and Competition on Large Scale Global Optimization. Nature Inspired Computation and Applications Laboratory, USTC, China, Technical Report, 2007.
[35] N. Vecek, M. Mernik, M. Crepinšek, A chess rating system for evolutionary algorithms: a new method for the comparison and ranking of evolutionary algorithms, Inform. Sci. 277 (2014) 656–679.
[36] H. Wang, Z. Wu, S. Rahnamayan, Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems, Soft Comput. 15 (2011) 2127–2140.
[37] M. Weber, F. Neri, V. Tirronen, Shuffle or update parallel differential evolution for large scale optimization, Soft Comput. 15 (2011) 2089–2107.
[38] D.L. Whitley, S. Rana, J. Dzubera, K.E. Mathias, Evaluating evolutionary algorithms, Artif. Intell. 85 (1996) 245–276.
[39] D. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1997) 67–82.
[40] Z. Yang, K. Tang, X. Yao, Scalability of generalized adaptive differential evolution for large-scale continuous optimization, Soft Comput. 15 (2011) 2141–2155.
[41] J.H. Zar, Biostatistical Analysis, fifth ed., Prentice Hall, 2009.
[42] S.Z. Zhao, P.N. Suganthan, Comprehensive comparison of convergence performance of optimization algorithms based on nonparametric statistical tests, in: Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2012, Brisbane, Australia, June 10–15.
[43] S.Z. Zhao, P.N. Suganthan, S. Das, Self-adaptive differential evolution with multi-trajectory search for large scale optimization, Soft Comput. 15 (2011) 2175–2185.