Large Scale Problems in Practice: The Effect of Dimensionality on the Interaction Among Variables

Fabio Caraffini[1,2], Ferrante Neri[1,2], and Giovanni Iacca[1,2(✉)]

[1] Centre for Computational Intelligence,
De Montfort University, The Gateway, Leicester LE1 9BH, UK
{fabio.caraffini,fneri}@dmu.ac.uk
[2] INCAS3, Dr. Nassaulaan 9, 9401 HJ Assen, The Netherlands
giovanni.iacca@gmail.com

AQ1

Abstract. This article performs a study on correlation between pairs of variables in dependence on the problem dimensionality. Two tests, based on Pearson and Spearman coefficients, have been designed and used in this work. In total, 86 test problems ranging between 10 and 1000 variables have been studied. If the most commonly used experimental conditions are used, the correlation between pairs of variables appears, from the perspective of the search algorithm, to consistently decrease. This effect is not due to the fact that the dimensionality modifies the nature of the problem but is a consequence of the experimental conditions: the computational feasibility of the experiments imposes an extremely shallow search in case of high dimensions. An exponential increase of budget and population with the dimensionality is still practically impossible. Nonetheless, since real-world application may require that large scale problems are tackled despite of the limited budget, an algorithm can quickly improve upon initial guesses if it integrates the knowledge that an apparent weak correlation between pairs of variables occurs, regardless the nature of the problem.

AQ2

Keywords: Large scale optimization · Covariance matrix · Correlation

1 Introduction

Dimensionality is a problem feature that is, in most cases, explicitly available when an optimization problem is formulated. It follows that a good algorithmic design should take into account the knowledge about the problem dimensionality to efficiently solve the problems at hand.

Optimization problems in many dimensions radically differ from low dimensional problems since the size of a domain grows exponentially with the number of dimensions. To remark this fact, let us consider a uni-dimensional decision space $\mathbf{D}$. Let $\mathbf{D}$ be a set composed of 100 points (candidate solutions). Let us consider now a function f defined over the set $\mathbf{D}$. Without loss of generality, let

G. Squillero and K. Sim (Eds.): EvoApplications 2017, Part I, LNCS 10199, pp. 1–17, 2017.
DOI: 10.1007/978-3-319-55849-3_41

us assume that there exists a solution $\mathbf{x}^* \in \mathbf{D}$ such that $f(\mathbf{x}^*)$ is minimal. Hence, in order to find the global minimum $\mathbf{x}^*$, an optimization algorithm needs at most 100 samples (or like it is often indicated in nature-inspired algorithms, fitness evaluations, see [1]). This problem would be very easy for a modern computer. On the other hand, if the problem is scaled up to two dimensions, there will be one optimum $\mathbf{x}^*$ in a space composed of $100^2 = 10000$ candidate solutions. If the problem is scaled up to 1000 dimensions, the optimum will be only one point in a space of 100^{1000} solutions. With an exhaustive search, the latter problem would be extremely hard to solve in a feasible time. Thus a specifically designed algorithm will be required to tackle it. In other words, since the decision space grows exponentially with the problem dimensionality, the detection of the optimal solution in high dimensions is like the search of a needle in a haystack, and requires some specific strategies.

In addition to that, the problem dimensionality affects not only the number of candidate solutions in the search space, but also other intrinsic features of the search space itself. For example, a unitary radius sphere in a 3-dimensional Euclidean space has area of the surface $S_2 = 4\pi$ and volume $V_3 = \frac{4}{3}\pi$. In the generic n-dimensional space, it can be easily proved that the ratio between volume and surface is $\frac{1}{n}$. This means that if we consider a unitary radius sphere in high dimensions and randomly sample some points within it, most of them will be located on its surface as its volume is a small fraction of it.

An optimization problem characterized by a high number of dimensions is known as Large Scale Optimization Problem (LSOP). Large scale problems can be hard to solve as some optimization algorithms that easily solve a problem in e.g. 30 dimensions can display a poor performance to solve the same problem scaled up to e.g. 300 dimensions. The deterioration in the performance of optimization algorithms as the dimensionality of the search space increases is commonly called "curse of dimensionality", see [2], and generally affects every kind of search logic. For example, several studies show that Differential Evolution (DE) and Particle Swarm Optimization (PSO) can easily display a poor performance when applied to solve LSOPs, see e.g. [2,3].

Furthermore, the dimensionality has a direct impact on the computational cost of the optimization, see [4]. In general, this is true because, due to the large decision space, a large budget is usually necessary to detect a solution with a high performance. Moreover, due to high dimensionality, algorithms which perform a search within the neighborhood of a candidate solution (e.g. Hooke-Jeeves Algorithm, [5]) might require a very large number of fitness evaluations at each step of the search, while population based algorithms are likely to either prematurely converge to suboptimal solutions, or stagnate due to an inability to generate new promising search directions. Other approaches that inspect the interaction between pairs or variables in order to perform an exploratory move, see e.g. [6], can be computationally onerous and in some cases, see e.g. [7], unacceptably expensive for modern computers.

A trivial but overlooked consideration regarding the scalability of optimization problems is that the parameter setting of the algorithm and the experimental

setup should take the dimensionality into account. At first, if a population-based algorithm is used, an exponential increase in the population size should be performed to keep the domain coverage constant. For example, if a DE algorithm is run with a population size of 30 individuals in 10 dimensions, to reproduce the same coverage in 50 dimensions, $30^{50-10} \approx 1.22 \times 10^{59}$ solutions would be needed. A population of this size is in practice impossible to use in the vast majority of problems. Hence, metaheuristics in high dimensions cover only a minimal part of the decision space.

A similar consideration can be done on the computational budget. Let us consider a DE algorithm run to solve a 10−dimensional problem with a budget of 50000 fitness evaluations. In order to explore/visit the same portion of decision space in 50 dimensions a budget of $50000^{40} \approx 9.1 \times 10^{187}$ fitness evaluations. Also this setting would be infeasible.

While in some cases scalability can simply be addressed by heuristic rules that scale up the algorithm parameters (e.g. by imposing that the computational budget is proportional to the problem dimensionality), as we have seen this strategy is not always possible, let alone efficient. Nevertheless, in real-world applications LSOPs must often be tackled efficiently in order to achieve a solution with a reasonable performance, e.g. in scheduling [8], chemical engineering [9,10], and in engineering design [11,12]. To address the aforementioned feasibility issues in terms of computational cost, several algorithms have been therefore proposed in the literature for handling this kind of problems.

In this paper, we present a study on the effect of dimensionality in optimization problems when the usual experimental conditions are set. Our purpose is to shed light on some specific characteristics that we consider especially interesting in large scale optimization and that, according to our empirical results, are common -to some extent- to most LSOPs. Among the features that can be used to analyze the fitness landscape in LSOPs, we focus on the correlation between pairs of problem variables. In particular, we study how the pairwise correlation changes in dependence on the dimensionality of the problem, in an attempt to address the research question: *On the basis of the usual algorithm and experimental setting, what happens to the correlation among the variables when the dimensionality grows?*

To address this question we illustrate a procedure to estimate the correlation between pairs of variables, an averaging technique to extract a unique measure that the describes the overall correlation among variables, and a sensitivity analysis of this measure with respect to the problem dimensionality. We applied the proposed analysis over a number of scalable problems commonly used in continuous optimization benchmarks, with dimensionality ranging between 10 and 1000 dimensions.

The remainder of this paper is organized in the following way. Section 2 shows successful strategies proposed in the literature to tackle LSOPs. Section 3 gives the implementation details of the procedure for estimating the correlation between pairs of variables. Section 4 shows the numerical results on a broad set of benchmark functions. Finally, Sect. 5 concludes this work.

2 Background: A Literature Review on Large Scale Optimization

In recent years, several modern metaheuristics have been proposed in order to tackle LSOPs, such as some modified versions of Ant Colony Optimization (ACO) [13] and Artificial Bee Colony Algorithm (ABC) [14]. In our view, the methods for tackling LSOPs can be roughly divided into two main categories:

- **Methods that intensively exploit promising search directions.** These algorithms with an apparent counterintuitive action, instead of exploring the large decision space, give up the search for the global optimum and use an intensive exploitation to improve upon a set of initial solutions to detect a solution with a high quality (regardless its optimality). Two popular ways to implement this approach have been proposed in the literature. The first way to achieve this aim is by using population-based algorithms with very small populations, see [15–18], or with a population that shrinks during the run, see [19–22]. The second way to achieve this aim is by using highly exploitative local search algorithms by combining them with other algorithms and integrating them within population based structures. In particular, a coordination of multiple local search components is used to tackle LSOPs in [23]. This logic, a part or a modification of it has been coupled and integrated within other algorithmic frameworks in [24–28]. It must be remarked that these algorithms tend to use a simple local search component that exploits the decision space by perturbing the candidate solution along the axes. Another interesting study belonging to this category has been presented in [29], where a modified version of Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is proposed for tackling separable problems. In this case, the proposed algorithm makes use of a diagonal matrix to determine the newly sampled points and then the search directions. Hence, this modified version of CMA-ES performs moves along the axes to solve separable problems. It was shown that this algorithmic scheme appeared especially promising in high dimensional cases.
- **Methods that decompose the search space.** Some other papers propose a technique, namely cooperative coevolution, originally defined in [30] and subsequently developed in other works, see e.g. [31,32]. The concept of the cooperative coevolution is to decompose a LSOP into a set of low-dimensional problems which can be separately solved and then recombined in order to compose the solution of the original problem. It is obvious that if the fitness function is separable, then the problem decomposition can be trivial, while for non-separable functions the problem decomposition can turn out to be a very difficult task. However, some techniques for performing the decomposition of non-separable functions have been developed, see [33]. Recently, cooperative coevolution procedures have been successfully integrated within DE frameworks for solving LSOPs, see e.g. [34–38]. Another very successful implementation of cooperative coevolution has been integrated within a PSO framework in [39].

A common denominator in these approaches is that the algorithm attempts to quickly achieve improvements by exploiting the search directions. In other words, since the budget is very limited and there is a large margin of improvement with respect to an initial sampling, every effective modern metaheuristic for LSOPs gives up the search for the global optimum and simply tries to enhance as much as possible upon an initial sampling.

3 Procedure for Estimating the Correlation Between Pairs of Variables

The proposed correlation estimation procedure is performed in two steps: (1) a preliminary sampling process, needed to sample a set of solutions in the search space and *evolve* them for a given number of generations; and (2) an estimation of the correlation among the variables in the final set of solutions obtained at the end of the evolutionary process performed in the first step. In this paper we use two different correlation measures, while the sampling mechanism is the same for both measures.

3.1 First Step: Covariance Matrix Adaptation Evolution Strategy

During the preliminary step, the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) with rank-μ-update and weighted recombination, see [40], is applied for $n \times 1000$ fitness evaluations. Briefly, the CMA-ES consists of sampling from a multivariate distribution λ points, computing their fitness values and updating the shape of the distribution in order to progressively adapt to the basins of attraction. The sampling rule of the individual k at the generation $g+1$ is given by:

$$x_k^{(g+1)} \sim \mathcal{N}\left(\langle \mathbf{x} \rangle_w^g, \left(\sigma^g\right)^2 \mathbf{C}^g\right) \tag{1}$$

where $\mathcal{N}\left(\mathbf{m}, \sigma^2 \mathbf{C}\right)$ is a multivariate normal distribution of mean $\mathbf{m}$, step size σ, and estimated covariance matrix $\mathbf{C}$. The mean value $\langle \mathbf{x} \rangle_w^g$ is a weighted sum of the μ candidate solutions ($\mu \leq \lambda$) displaying the best performance at the generation g (those individuals that are associated to the lowest fitness values $f(\mathbf{x})$). This vector corresponds to a recombination result, see [40] for details. At each g^{th} generation, the values of step size σ and covariance matrix $\mathbf{C}$ are updated. The two update rules are determined by a vector $\mathbf{p_c}$, named evolution path. The evolution path $\mathbf{p_c}$ is updated first, according to the following rule:

$$\mathbf{p_c}^{g+1} = (1 - c_c)\mathbf{p_c}^g + H_\sigma \sqrt{c_c\,(2 - c_c)} \frac{\sqrt{\mu_{eff}}}{\sigma^g} \left(\langle \mathbf{x} \rangle_w^{g+1} - \langle \mathbf{x} \rangle_w^g\right)$$

where $\mu_{eff} = \frac{1}{\sum_1^\mu w_i^2}$, c_c is a parameter, and H_σ is a discrete function that can take either the value 0 or 1. The step size is then updated according to the following rule:

$$\sigma^{g+1} = \sigma^g \exp\left(\frac{c_c}{d_c}\left(\frac{\|\mathbf{p_c}\|}{E\|(\mathcal{N}(\emptyset, \mathbf{I}))\|} - 1\right)\right)$$

where d_d is a damping factor, usually close to one, and $\|\dots\|$ indicates the 2-norm (see [41] for further explanations). Finally, $\mathbf{C}$ is updated according to the following rule:

$$\mathbf{C}^{g+1} = (1 - c_1 - c_\mu + c_s)\mathbf{C}^g + c_1 \mathbf{p_c}^{g+1} \left(\mathbf{p_c}^{g+1}\right)^T + c_\mu \sum_{i=1}^{\mu} w_i \left(\frac{\mathbf{x}_{i:\lambda}^{g+1} - \langle \mathbf{x} \rangle_w^g}{\sigma^g} \right) \left(\frac{\mathbf{x}_{i:\lambda}^{g+1} - \langle \mathbf{x} \rangle_w^g}{\sigma^g} \right)^T$$

where c_1, c_μ and c_s are learning parameters. At the end of each generation, the μ individuals displaying the best performance are selected and used to compute $\langle \mathbf{x} \rangle_w^{g+1}$. Implementation details about the CMA-ES structure and functioning can be found in [40,42,43].

According to the philosophy of CMA-ES, the matrix $\mathbf{C}$ evolves and reliably approximates the theoretical covariance matrix. A covariance matrix is a correlation matrix, i.e. a matrix that describes the correlation between pairs of variables and, at the same time, approximates the shape of the basins of attraction, i.e. those regions of the fitness landscape surrounding the fitness minima. In this way, CMA-ES samples new points from a distribution that adapts to the fitness landscape itself. In our tests, we empirically chose the CMA-ES budget ($n \times 1000$ fitness evaluations) so to have a reliable estimation of the covariance matrix, even though the convergence condition is likely still to be met. In other words, after this budget, the CMA-ES is likely to still sample points in a large portion of the decision space and not only a local basin of attraction. Due to the curse of dimensionality, this statement becomes progressively more true as the problem dimensionality grows (since the complexity grows exponentially while the CMA-ES budget is assigned by means of a linear formula).

3.2 Second Step: Correlation Estimation

Once the estimated covariance matrix $\mathbf{C}$ has been computed (i.e. after the given number of fitness evaluations), we calculate the pairwise correlation among the decision variables. Here we use, independently, two alternative correlation measures, namely the Pearson [44] and the Spearman correlation coefficients [45]. To calculate the Pearson coefficients, we take each element of the matrix $C_{i,j}$ and apply the following transformation:

$$\rho_{i,j} = \frac{C_{i,j}}{\sqrt{C_{i,i} C_{j,j}}}. \tag{2}$$

where $\rho_{i,j}$ define the Pearson correlation coefficients. These coefficients vary between -1 and 1 and measure the linear correlation between pairs of variables. When $\rho_{i,j} = 0$, there is no correlation at all between the i^{th} and j^{th} variables. When $|\rho_{i,j}| = 1$, there is a perfect correlation between the variables. More specifically, when $\rho_{i,j} = 1$, it means that to an increase of the i^{th} variable corresponds the same (linear) increase of the j^{th} variable; when $\rho_{i,j} = -1$, it means that to an increase of the i^{th} variable corresponds the same (linear) decrease of the

j^{th} variable. The matrix ρ composed of elements $\rho_{i,j}$ is the Pearson correlation matrix. The Pearson correlation matrix is more intuitive than the covariance matrix because its elements are limited and normalized within the $[-1, 1]$ interval, thus allowing an immediate interpretation. Since there is no interest, within this study, to distinguish between positive and negative correlation, the absolute value of the Pearson correlation matrix $|\rho|$ is computed. Moreover, since the correlation between variables is a symmetric relation and the self-correlation is the maximum possible correlation, the Pearson matrix has the following structure:

$$|\rho| = \begin{pmatrix} 1 & |\rho_{1,2}| & |\rho_{1,3}| & ... & |\rho_{1,n}| \\ X & 1 & |\rho_{2,3}| & ... & |\rho_{2,n}| \\ X & X & 1 & ... & |\rho_{3,n}| \\ ... & ... & ... & ... & ... \\ X & X & X & X & 1 \end{pmatrix}.$$

Thus, only $\frac{(n^2-n)}{2}$ elements of the matrix $|\rho|$ are of interest. In order to extract an index that performs an estimation of the average correlation among the variables, we simply average the elements of the matrix $|\rho|$:

$$\varsigma = \frac{2}{n^2 - n} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} |\rho_{i,j}|. \tag{3}$$

The Spearman correlation estimate consists of the following. By means of the covariance matrix $\mathbf{C}$, m points are sampled within the decision space. Considering that each point $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is a vector having n elements, these m points compose the following $m \times n$ matrix:

$$\mathbf{X} = \begin{pmatrix} x_{1,1} & x_{1,2} & x_{1,3} & ... & x_{1,n} \\ x_{2,1} & x_{2,2} & x_{2,3} & ... & x_{2,n} \\ ... & ... & ... & ... & ... \\ x_{m,1} & x_{m,2} & x_{m,3} & ... & x_{m,n} \end{pmatrix} = (\mathbf{X^1}, \mathbf{X^2}, \ldots, \mathbf{X^n})$$

where $\mathbf{X^j}$ is the generic j^{th} column vector of the matrix $\mathbf{X}$.

For each column vector, the elements are substituted with their ranking. More specifically, for the generic column vector $\mathbf{X^j}$ the lowest value is replaced with its ranking 1, the second lowest with 2, and so on until the highest value is replaced with n. If l elements have the same value, an average ranking is assigned. For example if three elements corresponding to the rank 3, 4, and 5 have the same value, the ranking 4 is assigned to all of them. This procedure can be seen as a matrix transformation that associates to the matrix $\mathbf{X}$ a new rank matrix $\mathbf{R}$ where the element $x_{i,j}$ is replaced with its rank $r_{i,j}$:

$$\mathbf{R} = \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & ... & r_{1,n} \\ r_{2,1} & r_{2,2} & r_{2,3} & ... & r_{2,n} \\ ... & ... & ... & ... & ... \\ r_{m,1} & r_{m,2} & r_{m,3} & ... & r_{m,n} \end{pmatrix} = (\mathbf{R^1}, \mathbf{R^2}, \ldots, \mathbf{R^n}).$$

From the rank matrix $\mathbf{R}$, a new matrix $\mathbf{T}$ is calculated by computing the Pearson correlation coefficients of the ranks. More specifically, the correlation between the i^{th} and j^{th} variables is given by:

$$\tau_{i,j} = \frac{\sum_{k=1}^{m}\left(r_{k,i} - \bar{\mathbf{R}^{\mathbf{i}}}\right)\sum_{k=1}^{m}\left(r_{k,j} - \bar{\mathbf{R}^{\mathbf{j}}}\right)}{\sqrt{\sum_{k=1}^{m}\left(r_{k,i} - \bar{\mathbf{R}^{\mathbf{i}}}\right)^2 \sum_{k=1}^{m}\left(r_{k,j} - \bar{\mathbf{R}^{\mathbf{j}}}\right)^2}} \tag{4}$$

where $\bar{\mathbf{R}^{\mathbf{i}}}$ and $\bar{\mathbf{R}^{\mathbf{j}}}$ are the mean values of the i^{th} and j^{th} column vectors, respectively. $\tau_{i,i} = 1$ define the Spearman coefficients.

Considering that $\forall i, j$, it results that $\tau_{i,i} = 1$ and $\tau_{i,j} = \tau_{j,i}$, the matrix $\mathbf{T}$ is symmetric and displays unitary diagonal elements. Since, analogous to the Pearson coefficient, we are not interested in the sign of the correlation, the absolute value of the matrix $\mathbf{T}$ is calculated:

$$|\mathbf{T}| = \begin{pmatrix} 1 & |\tau_{1,2}| & |\tau_{1,3}| & ... & |\tau_{1,n}| \\ X & 1 & |\tau_{2,3}| & ... & |\tau_{2,n}| \\ X & X & 1 & ... & |\tau_{3,n}| \\ ... & ... & ... & ... & ... \\ X & X & X & X & 1 \end{pmatrix}.$$

We then compute the average Spearman correlation index φ as the average value of the $\frac{(n^2-n)}{2}$ elements of the matrix $\mathbf{T}$ under consideration:

$$\varphi = \frac{2}{n^2 - n}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}|\tau_{i,j}|. \tag{5}$$

Before discussing the results, it is worth mentioning why we use two different correlation coefficients here. As explained in [46], the Pearson coefficient is more accurate if the pairwise correlation can be approximated to be linear. This circumstance realistically occurs in several -but not all- cases, in which the Pearson coefficient is reliable [47]. On the other hand, the Spearman coefficient is not a measure of the linear correlation between two variables, but rather it simply assesses how well the relationship between two variables can be described using a monotonic (not necessarily linear) function. Moreover the Spearman coefficient is non-parametric (distribution free), i.e. it does not require any assumption on the statistical process, it is less sensitive to outliers than the Pearson coefficient, but its calculation is computationally more expensive.

4 Numerical Results

We initially tested the procedure illustrated above over the 19 scalable test problems introduced in the Test Suite for the *Special Issue of Soft Computing on Scalability of Evolutionary Algorithms and other Metaheuristics for Large Scale Continuous Optimization Problems* [48], hereafter SISC2010. We calculated both Pearson and Spearman correlation coefficients over these 19 test problems in 10, 30, 50, 100, 500 and 1000 dimensions. To obtain robust correlation indications (which are affected by the stochastic nature of the sampling process), we calculated each aggregate index (ς and φ) 50 times per problem/dimension. In the following, we will explicitly refer to the dimensionality of the corresponding

problem every time we will mention an index. For example, to indicate the ς value for a problem in 30 dimensions we will write ς_{30n}.

In order to find the optimal number of samples to be drawn from the distribution to have a reliable calculation of correlation, we performed a preliminary experiment on SISC2010 in the aforementioned dimensionality values, with sample sizes proportional to the dimensionality of the problem. We tested four configurations, namely n, $5n$, $10n$ and $100n$, and for each of them we calculated the corresponding index values ς and φ. We observed that, regardless of the problem dimension, a set of 100 points, for the Pearson index, and 1000, for the Spearman index, provide an index as stable and reliable as that obtained by a higher number of points.

Following the indications of this preliminary experiment, we set the population size for CMA-ES equal to 100, which also allowed us to keep the experimental setup quite simple and computationally affordable. In case of Spearman index, the final population is sampled 10 times.

Table 1 displays the average correlation indices (calculated over the 50 runs available) and the corresponding standard deviation obtained on the entire SISC2010 benchmark. As it can immediately be observed, both the proposed indices appear to be closely related to the problem dimensionality. More specifically, regardless of the nature of the problem, the correlation amongst variables appear to decay with the dimensionality. All the problems display the maximum values of Pearson and Spearman indices in low dimensions, while these indices tend to take small values in large scale cases being nearly null in 1000 dimensions.

Moreover, we can observe that the correlation between pairs of variables appears somehow related to the separability of the problem, i.e. it tends to be lower when the problem is separable, as also noted in [49]. Although we admit that the concepts of correlation and separability are not strictly linked, we have conjectured the following explanation for the relation between these two concepts: since separable functions in n variables can be expressed as the sum of n functions in one variable, the problem is somehow characterized by a low correlation among variables. However, the opposite is not necessarily true: a low correlation among variables does not implicate problem separability. Some non-separable problems can still be characterized by a low variable correlation.

In order to confirm that the obtained results are not biased by the chosen testbed, we performed the same tests also over the *2013 IEEE Congress on Evolutionary Computation* (CEC2013) testbed, see [50] and the *2010 Black Box Optimization Benchmarking* (BBOB2010) testbed, see [51]. The first is scalable only for a limited amount of dimensionality values, that is 10, 30, and 50 dimensions. The second testbed is scalable in 10, 30, 50 and 100 dimensions. We performed the tests on the two testbeds over all problems in all the available dimensionality values, again with each test repeated 50 times. Numerical results on the CEC2013 testbed, showing the correlation indices averaged over the 50 runs available, are reported in Table 2 for both Pearson and Spearman indices. Numerical results on the BBOB2010 testbed are given in Table 3. On both testbeds and indices, it can be observed the same trend seen for SISC2010.

Table 1. Average Pearson (ς) and Spearman (φ) correlation indices $\pm$ standard deviation values for SISC2010 over increasing dimensionality values

	ς_{10n}	ς_{30n}	ς_{50n}	ς_{100n}	ς_{500n}	ς_{1000n}	Separable
f_1	0.054 ± 0.006	0.032 ± 0.001	0.025 ± 0.001	0.015 ± 0.001	0.002 ± 0.000	0.001 ± 0.000	YES
f_2	0.068 ± 0.008	0.043 ± 0.002	0.034 ± 0.001	0.021 ± 0.001	0.014 ± 0.0015	0.021 ± 0.001	–
f_3	0.178 ± 0.077	$0.072 \pm .0190$	0.035 ± 0.002	0.020 ± 0.002	0.002 ± 0.000	0.001 ± 0.000	–
f_4	0.058 ± 0.014	0.036 ± 0.005	0.026 ± 0.001	0.019 ± 0.002	0.004 ± 0.000	0.002 ± 0.000	YES
f_5	0.053 ± 0.007	0.033 ± 0.001	0.026 ± 0.001	0.017 ± 0.000	0.003 ± 0.000	0.001 ± 0.000	–
f_6	0.058 ± 0.010	0.038 ± 0.002	0.030 ± 0.008	0.017 ± 0.004	0.003 ± 0.001	0.002 ± 0.000	YES
f_7	0.059 ± 0.018	0.033 ± 0.001	0.035 ± 0.001	0.019 ± 0.001	0.003 ± 0.000	0.001 ± 0.000	YES
f_8	0.146 ± 0.008	0.069 ± 0.002	0.067 ± 0.002	0.053 ± 0.004	0.007 ± 0.002	0.003 ± 0.000	–
f_9	0.508 ± 0.428	0.073 ± 0.084	0.069 ± 0.064	0.094 ± 0.039	0.040 ± 0.015	0.025 ± 0.003	–
f_{10}	0.051 ± 0.004	0.033 ± 0.001	0.024 ± 0.001	0.016 ± 0.003	0.004 ± 0.000	0.002 ± 0.000	–
f_{11}	0.276 ± 0.272	0.092 ± 0.056	0.097 ± 0.066	0.068 ± 0.040	0.037 ± 0.0031	0.021 ± 0.006	–
f_{12}	0.109 ± 0.029	0.055 ± 0.008	0.041 ± 0.006	0.032 ± 0.009	0.008 ± 0.0006	0.005 ± 0.001	–
f_{13}	0.241 ± 0.077	0.075 ± 0.021	0.058 ± 0.003	0.046 ± 0.004	0.014 ± 0.0006	0.010 ± 0.001	–
f_{14}	0.091 ± 0.010	0.055 ± 0.005	0.040 ± 0.004	0.031 ± 0.007	0.010 ± 0.0015	0.007 ± 0.001	–
f_{15}	0.056 ± 0.008	0.040 ± 0.002	0.029 ± 0.000	0.025 ± 0.010	0.008 ± 0.0017	0.006 ± 0.001	–
f_{16}	0.094 ± 0.011	0.092 ± 0.024	0.084 ± 0.028	0.048 ± 0.013	0.017 ± 0.0021	0.013 ± 0.004	–
f_{17}	0.206 ± 0.121	0.144 ± 0.050	0.078 ± 0.011	0.061 ± 0.013	0.024 ± 0.0021	0.015 ± 0.002	–
f_{18}	0.226 ± 0.224	0.074 ± 0.024	0.049 ± 0.021	0.053 ± 0.034	0.036 ± 0.0099	0.022 ± 0.001	–
f_{19}	0.066 ± 0.006	0.063 ± 0.007	0.045 ± 0.007	0.027 ± 0.012	0.006 ± 0.0006	0.004 ± 0.001	–
	φ_{10n}	φ_{30n}	φ_{50n}	φ_{100n}	φ_{500n}	φ_{1000n}	Separable
f_1	0.093 ± 0.011	0.085 ± 0.003	0.084 ± 0.002	0.082 ± 0.001	0.080 ± 0.001	0.014 ± 0.000	YES
f_2	0.099 ± 0.010	0.091 ± 0.003	0.089 ± 0.002	0.083 ± 0.001	0.082 ± 0.001	0.016 ± 0.001	–
f_3	0.219 ± 0.062	0.113 ± 0.018	0.092 ± 0.005	0.084 ± 0.002	0.081 ± 0.002	0.014 ± 0.000	–
f_4	0.127 ± 0.017	0.091 ± 0.004	0.085 ± 0.003	0.083 ± 0.001	0.080 ± 0.001	0.014 ± 0.000	YES
f_5	0.103 ± 0.012	0.086 ± 0.002	0.083 ± 0.001	0.081 ± 0.001	0.080 ± 0.001	0.014 ± 0.000	–
f_6	0.096 ± 0.009	0.089 ± 0.004	0.085 ± 0.004	0.082 ± 0.001	0.080 ± 0.001	0.014 ± 0.000	YES
f_7	0.103 ± 0.018	0.086 ± 0.004	0.086 ± 0.002	0.083 ± 0.001	0.080 ± 0.001	0.014 ± 0.000	YES
f_8	0.186 ± 0.019	0.116 ± 0.002	0.116 ± 0.003	0.097 ± 0.003	0.081 ± 0.003	0.014 ± 0.000	–
f_9	0.675 ± 0.417	0.128 ± 0.061	0.112 ± 0.046	0.127 ± 0.029	0.096 ± 0.029	0.022 ± 0.002	–
f_{10}	0.097 ± 0.010	0.088 ± 0.004	0.084 ± 0.002	0.082 ± 0.001	0.081 ± 0.001	0.014 ± 0.000	–
f_{11}	0.577 ± 0.404	0.187 ± 0.101	0.112 ± 0.057	0.115 ± 0.010	0.098 ± 0.030	0.022 ± 0.001	–
f_{12}	0.160 ± 0.020	0.102 ± 0.007	0.096 ± 0.005	0.093 ± 0.007	0.084 ± 0.007	0.017 ± 0.001	–
f_{13}	0.237 ± 0.069	0.130 ± 0.014	0.130 ± 0.006	0.085 ± 0.004	0.089 ± 0.004	0.019 ± 0.001	–
f_{14}	0.123 ± 0.011	0.101 ± 0.008	0.098 ± 0.005	0.093 ± 0.004	0.087 ± 0.004	0.018 ± 0.002	–
f_{15}	0.105 ± 0.015	0.089 ± 0.036	0.089 ± 0.003	0.084 ± 0.003	0.081 ± 0.003	0.022 ± 0.000	–
f_{16}	0.132 ± 0.014	0.142 ± 0.030	0.128 ± 0.021	0.105 ± 0.008	0.089 ± 0.008	0.020 ± 0.003	–
f_{17}	0.210 ± 0.138	0.189 ± 0.067	0.123 ± 0.014	0.109 ± 0.011	0.088 ± 0.011	0.018 ± 0.001	–
f_{18}	0.386 ± 0.229	0.134 ± 0.067	0.098 ± 0.015	0.106 ± 0.025	0.096 ± 0.025	0.020 ± 0.002	–
f_{19}	0.108 ± 0.011	0.102 ± 0.009	0.102 ± 0.004	0.085 ± 0.009	0.081 ± 0.009	0.014 ± 0.001	–

Finally, we have taken into account the testbed for *Large Scale Global Optimization* introduced at CEC2013 (CEC2013-LSGO), for a further check. This testbed is available only at 1000 dimensions and its results are reported in Table 4. Again, the displayed indices have been averaged over 50 independent runs. As shown in Table 4, the results reported for the other testbeds are further confirmed. In this large scale case, all the indices tend to take a low value regardless of the fact the corresponding problem is separable or non-separable.

Table 2. Average Pearson (ς) and Spearman (φ) correlation indices $\pm$ standard deviation values for CEC2013 over increasing dimensionality values

	$\varsigma 10n$	$\varsigma 30n$	$\varsigma 50n$	$\varphi 10n$	$\varphi 30n$	$\varphi 50n$	Separable
f_1	0.054 ± 0.007	0.032 ± 0.001	0.027 ± 0.001	0.099 ± 0.010	0.085 ± 0.005	0.084 ± 0.001	YES
f_2	0.563 ± 0.036	0.256 ± 0.025	0.238 ± 0.024	0.543 ± 0.053	0.293 ± 0.038	0.242 ± 0.030	–
f_3	0.271 ± 0.147	0.116 ± 0.019	0.059 ± 0.025	0.267 ± 0.108	0.127 ± 0.023	0.112 ± 0.027	–
f_4	0.110 ± 0.028	0.085 ± 0.011	0.091 ± 0.009	0.154 ± 0.061	0.121 ± 0.006	0.121 ± 0.013	–
f_5	0.072 ± 0.010	0.058 ± 0.008	0.045 ± 0.004	0.113 ± 0.015	0.100 ± 0.006	0.091 ± 0.007	YES
f_6	0.601 ± 0.222	0.315 ± 0.045	0.164 ± 0.010	0.543 ± 0.224	0.305 ± 0.075	0.187 ± 0.015	–
f_7	0.303 ± 0.208	0.100 ± 0.019	0.060 ± 0.006	0.176 ± 0.268	0.117 ± 0.013	0.098 ± 0.005	–
f_8	0.413 ± 0.061	0.142 ± 0.011	0.076 ± 0.004	0.409 ± 0.086	0.158 ± 0.010	0.108 ± 0.005	–
f_9	0.325 ± 0.106	0.119 ± 0.020	0.076 ± 0.013	0.372 ± 0.178	0.150 ± 0.020	0.107 ± 0.003	–
f_{10}	0.187 ± 0.010	0.093 ± 0.001	0.065 ± 0.001	0.201 ± 0.021	0.120 ± 0.003	0.103 ± 0.004	–
f_{11}	0.081 ± 0.047	0.038 ± 0.007	0.028 ± 0.002	0.132 ± 0.044	0.087 ± 0.003	0.084 ± 0.002	YES
f_{12}	0.277 ± 0.075	0.085 ± 0.004	0.055 ± 0.001	0.257 ± 0.051	0.115 ± 0.007	0.097 ± 0.004	–
f_{13}	0.235 ± 0.068	0.113 ± 0.029	0.066 ± 0.009	0.261 ± 0.066	0.121 ± 0.025	0.106 ± 0.005	–
f_{14}	0.057 ± 0.011	0.037 ± 0.010	0.036 ± 0.010	0.093 ± 0.011	0.090 ± 0.013	0.092 ± 0.006	–
f_{15}	0.194 ± 0.025	0.091 ± 0.005	0.065 ± 0.006	0.205 ± 0.031	0.119 ± 0.011	0.100 ± 0.010	–
f_{16}	0.435 ± 0.110	0.345 ± 0.137	0.273 ± 0.162	0.354 ± 0.145	0.285 ± 0.022	0.245 ± 0.093	–
f_{17}	0.210 ± 0.062	0.099 ± 0.015	0.051 ± 0.009	0.197 ± 0.069	0.117 ± 0.014	0.094 ± 0.005	–
f_{18}	0.283 ± 0.032	0.120 ± 0.016	0.082 ± 0.021	0.264 ± 0.038	0.150 ± 0.019	0.115 ± 0.015	–
f_{19}	0.255 ± 0.043	0.094 ± 0.011	0.073 ± 0.019	0.288 ± 0.059	0.124 ± 0.006	0.105 ± 0.010	–
f_{20}	0.360 ± 0.127	0.140 ± 0.006	0.076 ± 0.003	0.319 ± 0.046	0.156 ± 0.008	0.109 ± 0.006	–
f_{21}	0.075 ± 0.011	0.032 ± 0.002	0.031 ± 0.008	0.107 ± 0.012	0.088 ± 0.004	0.085 ± 0.002	–
f_{22}	0.058 ± 0.011	0.046 ± 0.017	0.045 ± 0.010	0.100 ± 0.014	0.089 ± 0.009	0.091 ± 0.005	YES
f_{23}	0.252 ± 0.147	0.097 ± 0.011	0.087 ± 0.052	0.219 ± 0.062	0.130 ± 0.029	0.122 ± 0.011	–
f_{24}	0.224 ± 0.074	0.067 ± 0.016	0.057 ± 0.013	0.262 ± 0.131	0.110 ± 0.018	0.094 ± 0.010	–
f_{25}	0.218 ± 0.122	0.115 ± 0.010	0.065 ± 0.008	0.197 ± 0.048	0.138 ± 0.010	0.102 ± 0.004	–
f_{26}	0.253 ± 0.087	0.068 ± 0.009	0.082 ± 0.069	0.301 ± 0.010	0.152 ± 0.011	0.150 ± 0.075	–
f_{27}	0.142 ± 0.093	0.082 ± 0.032	0.065 ± 0.021	0.138 ± 0.077	0.118 ± 0.032	0.105 ± 0.011	–
f_{28}	0.084 ± 0.063	0.033 ± 0.001	0.028 ± 0.013	0.118 ± 0.012	0.086 ± 0.003	0.085 ± 0.002	–

For all the 86 problems considered in this study, it appears clear that the correlation amongst variables (both Pearson and Spearman indices) tends to decay when the dimensionality increases. As a general trend, optimization problems with at least 100 dimensions seem characterized by a weak correlation. Optimization problems in 500 and 1000 dimensions show a nearly null correlation amongst the variables.

The phenomenon of the decrease of the correlation indices when the dimensionality increases is depicted in Figs. 1 and 2. It can be observed that all the trends decrease towards zero. In addition, we noticed that while for separable functions the indices have comparatively low values already in 10 dimensions and further become smaller with the increase of dimensionality, non-separable functions are characterized by a major drop in the correlation indices when the dimensionality increases (see Figs. 1 and 2).

Thus, our experimental study suggests that large scale optimization problems, regardless of the specific problem, have a lot in common with each other in terms of correlation among the variables and that all the LSOPs appear always

Table 3. Average Pearson (ς) and Spearman (φ) correlation indices $\pm$ standard deviation values for BBOB2010 over increasing dimensionality values

	ς_{10n}	ς_{30n}	ς_{50n}	ς_{100n}	φ_{10n}	φ_{30n}	φ_{50n}	φ_{100n}	Separable
f_1	0.054 ± 0.004	0.045 ± 0.035	0.028 ± 0.001	0.015 ± 0.000	0.095 ± 0.009	0.087 ± 0.003	0.084 ± 0.002	0.082 ± 0.001	YES
f_2	0.051 ± 0.006	0.034 ± 0.002	0.028 ± 0.001	0.015 ± 0.000	0.096 ± 0.008	0.087 ± 0.004	0.084 ± 0.002	0.081 ± 0.001	YES
f_3	0.054 ± 0.007	0.045 ± 0.040	0.030 ± 0.012	0.015 ± 0.000	0.096 ± 0.009	0.087 ± 0.003	0.084 ± 0.001	0.082 ± 0.001	YES
f_4	0.055 ± 0.007	0.045 ± 0.002	0.027 ± 0.001	0.015 ± 0.000	0.090 ± 0.008	0.087 ± 0.004	0.085 ± 0.002	0.081 ± 0.001	YES
f_5	0.058 ± 0.004	0.035 ± 0.002	0.027 ± 0.001	0.015 ± 0.000	0.092 ± 0.011	0.086 ± 0.003	0.084 ± 0.002	0.082 ± 0.001	YES
f_6	0.051 ± 0.006	0.042 ± 0.030	0.031 ± 0.015	0.015 ± 0.000	0.099 ± 0.016	0.084 ± 0.004	0.083 ± 0.002	0.082 ± 0.001	–
f_7	0.053 ± 0.005	0.033 ± 0.002	0.027 ± 0.001	0.015 ± 0.000	0.099 ± 0.013	0.087 ± 0.004	0.086 ± 0.001	0.081 ± 0.001	–
f_8	0.055 ± 0.006	0.034 ± 0.002	0.028 ± 0.001	0.015 ± 0.000	0.094 ± 0.010	0.086 ± 0.003	0.085 ± 0.002	0.081 ± 0.001	–
f_9	0.054 ± 0.005	0.039 ± 0.015	0.027 ± 0.001	0.015 ± 0.000	0.099 ± 0.010	0.089 ± 0.004	0.085 ± 0.001	0.082 ± 0.000	–
f_{10}	0.055 ± 0.007	0.034 ± 0.001	0.028 ± 0.001	0.015 ± 0.000	0.102 ± 0.018	0.087 ± 0.003	0.087 ± 0.008	0.082 ± 0.001	–
f_{11}	0.051 ± 0.005	0.043 ± 0.034	0.029 ± 0.008	0.015 ± 0.000	0.094 ± 0.011	0.087 ± 0.003	0.085 ± 0.001	0.082 ± 0.001	–
f_{12}	0.053 ± 0.005	0.034 ± 0.002	0.028 ± 0.001	0.015 ± 0.000	0.093 ± 0.014	0.088 ± 0.003	0.085 ± 0.002	0.082 ± 0.001	–
f_{13}	0.055 ± 0.007	0.035 ± 0.002	0.027 ± 0.001	0.015 ± 0.000	0.095 ± 0.011	0.086 ± 0.004	0.083 ± 0.002	0.081 ± 0.001	–
f_{14}	0.054 ± 0.007	0.034 ± 0.002	0.027 ± 0.001	0.015 ± 0.000	0.097 ± 0.009	0.088 ± 0.004	0.084 ± 0.001	0.082 ± 0.001	–
f_{15}	0.053 ± 0.007	0.035 ± 0.001	0.028 ± 0.001	0.015 ± 0.000	0.097 ± 0.008	0.086 ± 0.003	0.085 ± 0.001	0.082 ± 0.001	–
f_{16}	0.051 ± 0.004	0.054 ± 0.046	0.027 ± 0.001	0.015 ± 0.000	0.093 ± 0.013	0.085 ± 0.003	0.084 ± 0.002	0.082 ± 0.001	–
f_{17}	0.051 ± 0.004	0.035 ± 0.002	0.027 ± 0.001	0.015 ± 0.000	0.092 ± 0.013	0.087 ± 0.004	0.084 ± 0.003	0.082 ± 0.001	–
f_{18}	0.052 ± 0.006	0.039 ± 0.012	0.028 ± 0.001	0.015 ± 0.000	0.100 ± 0.012	0.089 ± 0.004	0.085 ± 0.003	0.082 ± 0.001	–
f_{19}	0.052 ± 0.006	0.034 ± 0.001	0.028 ± 0.001	0.015 ± 0.000	0.097 ± 0.008	0.087 ± 0.002	0.085 ± 0.002	0.082 ± 0.001	–
f_{20}	0.051 ± 0.006	0.034 ± 0.001	0.028 ± 0.001	0.015 ± 0.000	0.093 ± 0.009	0.086 ± 0.003	0.084 ± 0.001	0.082 ± 0.001	–
f_{21}	0.052 ± 0.006	0.034 ± 0.001	0.027 ± 0.001	0.015 ± 0.000	0.088 ± 0.008	0.087 ± 0.004	0.084 ± 0.002	0.082 ± 0.001	–
f_{22}	0.051 ± 0.004	0.038 ± 0.013	0.027 ± 0.001	0.015 ± 0.000	0.098 ± 0.010	0.086 ± 0.003	0.083 ± 0.002	0.082 ± 0.001	–
f_{23}	0.054 ± 0.007	0.035 ± 0.001	0.027 ± 0.001	0.015 ± 0.000	0.092 ± 0.011	0.091 ± 0.009	0.084 ± 0.003	0.082 ± 0.001	–
f_{24}	0.053 ± 0.005	0.033 ± 0.002	0.027 ± 0.001	0.015 ± 0.000	0.096 ± 0.008	0.086 ± 0.003	0.084 ± 0.002	0.082 ± 0.001	–

Table 4. Average Pearson (ς) and Spearman (φ) correlation indices $\pm$ standard deviation values for CEC2013-LSGO in 1000 dimensions

	ς_{1000n}	φ_{1000n}	Separable
f_1	0.015 ± 0.001	0.085 ± 0.001	YES
f_2	0.003 ± 0.000	0.080 ± 0.000	YES
f_3	0.002 ± 0.000	0.080 ± 0.000	YES
f_4	0.015 ± 0.001	0.085 ± 0.001	–
f_5	0.003 ± 0.000	0.080 ± 0.000	–
f_6	0.002 ± 0.000	0.080 ± 0.000	–
f_7	0.015 ± 0.001	0.085 ± 0.001	–
f_8	0.016 ± 0.001	0.085 ± 0.001	–
f_9	0.003 ± 0.000	0.080 ± 0.000	–
f_{10}	0.002 ± 0.000	0.080 ± 0.000	–
f_{11}	0.015 ± 0.001	0.086 ± 0.001	–
f_{12}	0.015 ± 0.002	0.085 ± 0.001	YES
f_{13}	0.015 ± 0.001	0.085 ± 0.001	–
f_{14}	0.015 ± 0.001	0.085 ± 0.002	–
f_{15}	0.015 ± 0.001	0.085 ± 0.001	–

characterized by uncorrelated variables. This fact has an effect on the design strategy since uncorrelated variables could be perturbed/optimised separately.

We are not concluding, however, that the nature of the problem changes with its dimensionality. In other words, we are not concluding that LSOPs are

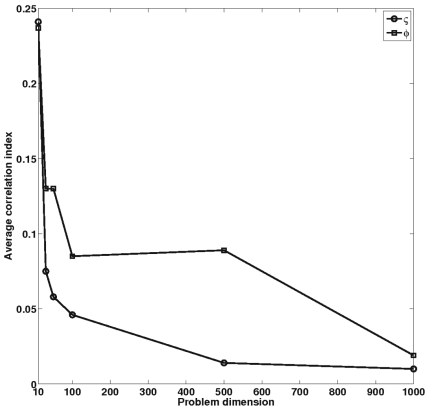

Fig. 1. Correlation indices for f_{13} of SISC2010

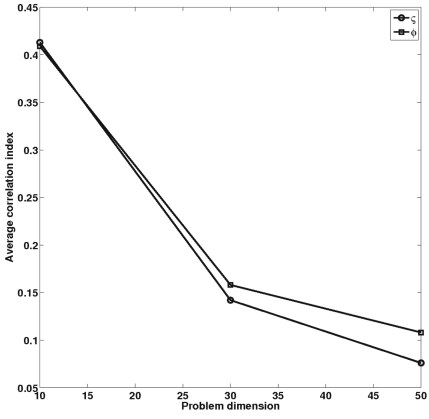

Fig. 2. Correlation indices for f_8 of CEC2013.

characterized by a weak correlation between pairs of variables. On the other hand, the experimental conditions imposed by the computational restrictions make LSOPs appear characterized by a low correlation since the search on this problems has to be much more shallow (i.e., it can cover a much smaller portion of the decision space) than the same problem in low dimensions. From a practical viewpoint, our conclusion is that since we know that in high dimensions we do not (and cannot) truly explore the decision space but we only attempt to improve upon some solutions with a very modest budget, the most effective way to quickly achieve an improvement would be to take into account this apparent weak correlation between pairs of variables. Although a further investigation is needed, the result of this study can be exploited during the algorithm design by employing exploitative techniques which perturb the variables one by one.

5 Conclusion

This paper proposed a technique based on two statical tests, based Pearson and Spearman correlation coefficients respectively, to measure the correlation between pairs of variables. These tests have been applied to measure the correlation between pairs of variables in different dimensionality scenarios. The standard experimental conditions used in the literature and popular competitions have been reproduced.

We noted that, in practice, such experimental conditions impose a growing shallowness of the search with the increase of dimensionality, i.e. only a very restricted portion of the decision space is explored in high dimensions. We observed that under these conditions, problems tend to appear, regardless of their nature, characterized by a weak correlation of variables.

Thus, if the budget is limited, a practically efficient approach could be, according to our conjecture, to avoid the use of exploratory components and simultaneous variations of multiple variables (diagonal moves). On the contrary, the exploitation of the search along each variable can enhance in the short term and for the limited budget the efficiency of the search. This conjecture is in accordance with the most popular and successful methods for large scale optimisation.

Further studies will propose specific algorithmic components which will make use of the knowledge gained in this study to leverage the (apparent) weak correlation between pairs of variables within their search logic.

References

1. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computation. Natural Computing Series. Springer, Berlin (2003)
2. van den Bergh, F., Engelbrecht, A.P.: A cooperative approach to particle swarm optimization. IEEE Trans. Evol. Comput. **8**(3), 225–239 (2004)
3. Neri, F., Tirronen, V.: Recent advances in differential evolution: a review and experimental analysis. Artif. Intell. Rev. **33**(1–2), 61–106 (2010)

4. Li, S., Wei, D.: Extremely high-dimensional feature selection via feature generating samplings. IEEE Trans. Cybern. **44**(6), 737–747 (2014)
5. Hooke, R., Jeeves, T.A.: Direct search solution of numerical and statistical problems. J. ACM **8**, 212–229 (1961)
6. Auger, A., Hansen, N.: A restart CMA evolution strategy with increasing population size. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1769–1776 (2005)
7. Molina, D., Lozano, M., Garcia-Martinez, C., Herrera, F.: Memetic algorithms for continuous optimization based on local search chains. Evol. Comput. **18**(1), 27–63 (2010)
8. Marchiori, E., Steenbeek, A.: An evolutionary algorithm for large scale set covering problems with application to airline crew scheduling. In: Cagnoni, S. (ed.) EvoWorkshops 2000. LNCS, vol. 1803, pp. 370–384. Springer, Heidelberg (2000). doi:10.1007/3-540-45561-2_36
9. Kononova, A.V., Hughes, K.J., Pourkashanian, M., Ingham, D.B.: Fitness diversity based adaptive memetic algorithm for solving inverse problems of chemical kinetics. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 2366–2373 (2007)
10. Kononova, A.V., Ingham, D.B., Pourkashanian, M.: Simple scheduled memetic algorithm for inverse problems in higher dimensions: application to chemical kinetics. In: Proceedings of the IEEE World Congress on Computational Intelligence, pp. 3906–3913 (2008)
11. Akay, B., Karaboga, D.: Artificial bee colony algorithm for large-scale problems and engineering design optimization. J. Intell. Manufact. **23**(4), 1001–1014 (2012)
12. Iacca, G., Caraffini, F., Neri, F.: Multi-strategy coevolving aging particle optimization. Int. J. Neural Syst. **24**(1), 1450008 (2014)
13. Korošec, P., Šilc, J.: The differential ant-stigmergy algorithm for large scale real-parameter optimization. In: Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T., Winfield, A.F.T. (eds.) ANTS 2008. LNCS, vol. 5217, pp. 413–414. Springer, Heidelberg (2008). doi:10.1007/978-3-540-87527-7_51
14. Fister, I., Jr., I.F., Brest, J., Zumer, V.: Memetic artificial bee colony algorithm for large-scale global optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1–8 (2012)
15. Parsopoulos, K.E.: Cooperative micro-differential evolution for high-dimensional problems. In: Proceedings of the Conference on Genetic and Evolutionary Computation, pp. 531–538 (2009)
16. Parsopoulos, K.E.: Cooperative micro-particle swarm optimization. In: Proceedings of the First ACM/SIGEVO Summit on Genetic and Evolutionary Computation, pp. 467–474. ACM (2009)
17. Rajasekhar, A., Das, S., Das, S.: Abc: a micro artificial bee colony algorithm for large scale global optimization. In: GECCO (Companion), pp. 1399–1400 (2012)
18. Dasgupta, S., Biswas, A., Das, S., Panigrahi, B.K., Abraham, A.: A micro-bacterial foraging algorithm for high-dimensional optimization. In: IEEE Congress on Evolutionary Computation, pp. 785–792 (2009)
19. Brest, J., Maučec, M.S.: Population size reduction for the differential evolution algorithm. Appl. Intell. **29**(3), 228–247 (2008)
20. Zamuda, A., Brest, J., Bošković, B., Žumer, V.: High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction. In: Proceedings of the IEEE World Congress on Computational Intelligence, pp. 2032–2039 (2008)

21. Iacca, G., Mallipeddi, R., Mininno, E., Neri, F., Suganthan, P.N.: Super-fit and population size reduction mechanisms in compact differential evolution. In: Proceedings of IEEE Symposium on Memetic Computing, pp. 21–28 (2011)
22. Brest, J., Maucec, M.S.: Self-adaptive differential evolution algorithm using population size reduction and three strategies. Soft Comput. **15**(11), 2157–2174 (2011)
23. Tseng, L.Y., Chen, C.: Multiple trajectory search for large scale global optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 3052–3059 (2008)
24. Zhao, S.Z., Suganthan, P.N., Das, S.: Self-adaptive differential evolution with multi-trajectory search for large-scale optimization. Soft Comput. **15**(11), 2175–2185 (2011)
25. Caraffini, F., Neri, F., Poikolainen, I.: Micro-differential evolution with extra moves along the axes. In: Proceedings of the IEEE Symposium Series on Computational Intelligence, pp. 46–53 (2013)
26. Iacca, G., Neri, F., Mininno, E., Ong, Y.S., Lim, M.H.: Ockham's razor in memetic computing: three stage optimal memetic exploration. Inf. Sci. **188**, 17–43 (2012)
27. Caraffini, F., Neri, F., Iacca, G., Mol, A.: Parallel memetic structures. Inf. Sci. **227**, 60–82 (2013)
28. Caraffini, F., Neri, F., Passow, B., Iacca, G.: Re-sampled inheritance search: high performance despite the simplicity. Soft Comput. **17**(12), 2235–2256 (2014)
29. Ros, R., Hansen, N.: A simple modification in CMA-ES achieving linear time and space complexity. In: Proceesdings of the Parallel Problem Solving in Nature, pp. 296–305 (2008)
30. Potter, M.A., Jong, K.A.: A cooperative coevolutionary approach to function optimization. In: Davidor, Y., Schwefel, H.-P., Männer, R. (eds.) PPSN 1994. LNCS, vol. 866, pp. 249–257. Springer, Heidelberg (1994). doi:10.1007/3-540-58484-6_269
31. Liu, Y., Zhao, Q.: Scaling up fast evolutionary programming with cooperative coevolution. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1101–1108 (2001)
32. Sofge, D., De Jong, K., Schultz, A.: A blended population approach to cooperative coevolution for decomposition of complex problems. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 413–418 (2002)
33. Potter, M.A., De Jong, K.: Cooperative coevolution: an architecture for evolving coadapted subcomponents. Evol. Comput. **8**(1), 1–29 (2000)
34. Shi, Y., Teng, H., Li, Z.: Cooperative co-evolutionary differential evolution for function optimization. In: Wang, L., Chen, K., Ong, Y.S. (eds.) ICNC 2005. LNCS, vol. 3611, pp. 1080–1088. Springer, Heidelberg (2005). doi:10.1007/11539117_147
35. Yang, Z., Tang, K., Yao, X.: Differential evolution for high-dimensional function optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 3523–3530 (2007)
36. Zamuda, A., Brest, J., Bošković, B., Žumer, V.: Large scale global optimization using differential evolution with self-adaptation and cooperative co-evolution. In: Proceedings of the IEEE World Congress on Computational Intelligence, pp. 3719–3726 (2008)
37. Olorunda, O., Engelbrecht, A.P.: Differential evolution in high-dimensional search spaces. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1934–1941 (2007)
38. Yang, Z., Tang, K., Yao, X.: Large scale evolutionary optimization using cooperative coevolution. Inf. Sci. **178**(15), 2985–2999 (2008)
39. Li, X., Yao, X.: Cooperatively coevolving particle swarms for large scale optimization. IEEE Trans. Evol. Comput. **16**(2), 210–224 (2012)

40. Hansen, N., Müller, S.D., Koumoutsakos, P.: Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evol. Comput. **11**(1), 1–18 (2003)
41. Hansen, N., Kern, S.: Evaluating the CMA evolution strategy on multimodal test functions. In: Yao, X., et al. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 282–291. Springer, Heidelberg (2004). doi:10.1007/978-3-540-30217-9_29
42. Hansen, N., Ostermeier, A.: Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In: Proceedings of the IEEE International Conference on Evolutionary Computation, pp. 312–317 (1996)
43. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evol. Comput. **9**(2), 159–195 (2001)
44. Pearson, K.: Mathematical contributions to the theory of evolution. XI. on the influence of natural selection on the variability and correlation of organs. Philos. Trans. Roy. Soc. Lon. Ser. A, Contain. Papers Math. Phys. Char. **200**, 1–66 (1903)
45. Spearman, C.: The proof and measurement of association between two things. Am. J. Psychol. **15**(1), 72–101 (1904)
46. Hauke, J., Kossowski, T.: Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data. Quaestiones Geographicae **2**, 87–93 (2011)
47. Cox, D.R., Hinkley, D.: Theoretical Statistics. Chapman & Hall, London (1974)
48. Lozano, M., Molina, D., Herrera, F.: Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems. Soft Comput. **15**(11), 2085–2087 (2011)
49. Caraffini, F., Neri, F., Picinali, L.: An analysis on separability for memetic computing automatic design. Inf. Sci. **265**, 1–22 (2014)
50. Liang, J.J., Qu, B.Y., Suganthan, P.N., Hernández-Díaz, A.G. : Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. Technical report 201212, Zhengzhou University and Nanyang Technological University, Zhengzhou China and Singapore (2013)
51. Hansen, N., Auger, A., Finck, S., Ros, R., et al.: Real-parameter black-box optimization benchmarking 2010: noiseless functions definitions. Technical report RR-6829, INRIA (2010)