Stat Comput (2016) 26:15–28
DOI 10.1007/s11222-014-9467-z

Optimal design of large-scale screening experiments: a critical look at the coordinate-exchange algorithm

Daniel Palhazi Cuervo · Peter Goos · Kenneth Sörensen

Received: 6 June 2013 / Accepted: 25 March 2014 / Published online: 11 May 2014

Abstract We focus on the D-optimal design of screening experiments involving main-effects regression models, especially with large numbers of factors and observations. We propose a new selection strategy for the coordinate-exchange algorithm based on an orthogonality measure of the design. Computational experiments show that this strategy finds better designs within an execution time that is 30 % shorter than other strategies. We also provide strong evidence that the use of the prediction variance as a selection strategy does not provide any added value in comparison to simpler selection strategies. Additionally, we propose a new iterated local search algorithm for the construction of D-optimal experimental designs. This new algorithm outperforms the original coordinate-exchange algorithm.

Keywords Optimal design of experiments · D-optimality criterion · Coordinate-exchange algorithm · Metaheuristic · Iterated local search

1 Introduction

One of the most important steps during the design phase of any product or process is the experimental study carried out to determine the impact of a set of potentially influential factors on a specific quality measure. The main objective of that study is to determine the relationship between a *response variable* and the settings of several *factors* or *experimental variables* that are assumed to affect it. Using (linear) regression techniques, it is possible to build a predictive model and identify the settings of the factors that produce the best value of the response variable. In order to design an experiment, it is necessary to fix the number of *observations* or *experimental runs* and the settings of the factors to be used at each observation. This should be done with the objective of maximizing the information produced by the experiment, in such a way that the regression model can be estimated as precisely as possible. There are several well-known standard experimental designs in the literature that achieve this goal. However, they cannot always be applied to the complex scenarios found in industry. The most common problems are the following: (I) the set of experimental variables includes both qualitative and quantitative variables, (II) there are special resource restrictions and the number of observations suggested by the standard designs cannot be afforded, (III) some factors are subject to one or more specific constraints, or (IV) a nonstandard regression model is required. Instead of forcing the experimenter to adapt the experiment in order to fit the available designs, the optimal design of experiments approach attempts to find the best possible design for each particular scenario. To accomplish this goal, it is necessary to solve a complex optimization problem the difficulty of which increases exponentially with the number of factors and observations (Welch 1982).

Several algorithms have been proposed for this problem in the past four decades. Most of these have the limitation of using candidate-set based procedures which drastically reduce the explored solution space and cannot be used for large experiments (Atkinson et al. 2007). In recent years, the advances in control and embedded systems have increased the capability of industries to carry out experiments with larger numbers of factors and observations. In the chemical and pharmaceutical industry, it is now common to find

D. Palhazi Cuervo (✉) · K. Sörensen · P. Goos
Faculty of Applied Economics, University of Antwerp, Antwerp, Belgium
e-mail: daniel.palhazicuervo@uantwerpen.be

P. Goos
Faculty of Bioscience Engineering, University of Leuven, Leuven, Belgium

advanced control systems that automatically adjust the configuration of the production process. Therefore, it is possible to program more complex experiments at an affordable cost. For example, screening experiments used in combinatorial chemistry for the discovery of new composite materials can include dozens of factors, each of which with multiple levels (Cawse 2003). This necessitates new algorithms that can generate optimal designs for larger scale experiments.

The use of exact algorithms has been the traditional approach to solve combinatorial optimization problems. These algorithms perform a systematic search of the solution space and guarantee to find the best possible solution. The main limitation of these algorithms is the fact that their execution time increases at an exponential rate, which means that they cannot solve large instances. More recently, heuristic algorithms have been shown to overcome this problem. Instead of performing an exhaustive search, these algorithms explore promising parts of the solution space by using a more "intelligent" search strategy. While the obtained solutions cannot be proven to be optimal, heuristic algorithms are generally able to find very good solutions in an acceptable computing time for instances that are intractable for exact algorithms. Most of the state-of-the-art heuristics to solve combinatorial optimization problems belong to the group of *metaheuristics*. A metaheuristic is a high-level problem-independent algorithmic framework that provides a set of guidelines or strategies to develop heuristic optimization algorithms (Sörensen and Glover 2013). Several metaheuristics are available in the literature. The family of local search algorithms explore adjacent portions of the solution space in order to find better solutions; simulated annealing and tabu search are the most important members of this family. Other metaheuristics enhance the performance of local search algorithms by applying different mechanisms to diversify the exploration process; for instance, iterated local search and the greedy randomized adaptive search procedure or GRASP. Other metaheuristics are inspired by natural processes that are not directly related to the field of optimization. Genetic algorithms, for example, are based on Darwin's natural selection theory of evolution, whereas ant colony optimization algorithms are based on the behavior of ants during the transportation of food to their colonies. For a more detailed explanation and classification of metaheuristics, see Michalewicz and Fogel (2004); Talbi (2009).

Within the field of optimal experimental designs, the coordinate-exchange algorithm (CEA) proposed by Meyer and Nachtsheim (1995) is viewed as the first algorithm that employed a different approach than the traditional point-exchange algorithms to overcome their limitations. Since the CEA does not use a set of candidate points, the portion of the solution space that is explored is not restricted. Nevertheless, the main limitation of the CEA is its tendency to get trapped in locally optimal designs, especially when dealing with experiments with large numbers of factors and observations. A strategy that is commonly used to overcome this issue is to execute the algorithm several times, each time starting from a different initial design (Jones 2008). Even though this restart strategy increases the probability of finding the global optimum design, it is not effective for the design of large-scale experiments.

In this paper, we introduce a new strategy for the CEA to select the coordinates to be exchanged. The objective is to start the exploration of the design matrix in an area where an exchange is likely to produce an improvement, and consequently, avoid unnecessary explorations of unpromising areas. We perform a thorough comparison of several selection strategies in order to find which one produces the best algorithm performance. Additionally, we also propose an iterated local search (ILS) metaheuristic for the construction of optimal designs. This algorithm has been shown to better avoid locally optimal solutions than the restart strategy (Lourenco et al. 2003). Instead of restarting the search process from scratch every time a locally optimal solution is found, the ILS algorithm applies a perturbation operator to the solution in order to better explore the surrounding solution space. We carried out several computational experiments in order to tune the parameters of our ILS algorithm and compare its performance to algorithms that are available in the literature and in commercial software packages.

This paper is organized as follows. We describe the optimal experimental design problem and its mathematical formulation in Sect. 2. We provide a brief review of the existing algorithms for the construction of optimal experimental designs in Sect. 3. This review involves a new classification based on the structural characteristics of the algorithms. We discuss the original selection strategy of the CEA in Sect. 4, where we also propose a new strategy based on an orthogonality measure of the design and compare it to other selection strategies. We propose a new ILS algorithm for the generation of optimal designs of experiments in Sect. 5. In Sect. 6, we show the results of a statistical analysis that was performed to tune the parameters of the ILS. We compare the performance of the ILS and the CEA found in a statistical software package in Sect. 7. We present our final conclusions in Sect. 8.

2 Problem description and mathematical formulation

Let v be the number of factors and n the number of observations in the experiment. Consider the main-effects linear regression model given by the expression $y_i = \beta_0 + \sum_{j=1}^{v} \beta_j x_{ij} + \varepsilon_i$, where y_i is the value of the response variable at the i-th observation, β_0 is the intercept of the regression model, β_j is the unknown regression parameter of the j-th factor, x_{ij} is the setting of the j-th experimental factor

used in the i-th run, and ε_i is a random error term added to the model to capture the random variation each observation is subject to. The collection of settings of the factors at the i-th experimental run is denoted by the column vector $\boldsymbol{x_i}$, and is referred to as the design point or treatment corresponding to the i-th observation. The model can be expressed in matrix notation as $\boldsymbol{Y} = \boldsymbol{X\beta} + \boldsymbol{\varepsilon}$, where $\boldsymbol{Y}$ is the $n \times 1$ column vector of responses, $\boldsymbol{X}$ is the design matrix composed of n rows and $(v+1)$ columns and $\boldsymbol{\beta}$ is the $(v+1) \times 1$ column vector that contains the unknown regression parameters. Each row of the design matrix is formed by the model expansion $\boldsymbol{f}'(\boldsymbol{x_i}) = (1, x_{i1}, x_{i2}, \ldots, x_{iv})$ in order to include the intercept of the regression model. The error terms are assumed to be independent and identically distributed random variables with mean zero and, with no loss of generality, variance $\sigma^2 = 1$. The ordinary least squares (OLS) estimator for the unknown model parameters is $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{Y}$ (Atkinson et al. 2007). The variance–covariance matrix can be expressed as $\text{var}(\hat{\boldsymbol{\beta}}) = (\boldsymbol{X}'\boldsymbol{X})^{-1}$, where $\boldsymbol{X}'\boldsymbol{X}$ is the $(v+1) \times (v+1)$ symmetric information matrix, which can be computed as $\sum_{i=1}^{n} \boldsymbol{f}(\boldsymbol{x_i})\boldsymbol{f}'(\boldsymbol{x_i})$.

The problem of constructing optimal designs consists of selecting n design points (or rows $\boldsymbol{f}'(\boldsymbol{x_i})$) in order to optimize a function of the information matrix; the criterion used to evaluate the designs defines the objective function to be maximized or minimized. The D-optimality criterion is the most commonly used objective function in practice; it minimizes the generalized variance of the parameter estimators. This is accomplished by minimizing the determinant of the estimator's variance–covariance matrix or, equivalently, maximizing the determinant $|\boldsymbol{X}'\boldsymbol{X}|$ of the information matrix. Since the determinant value exponentially grows with the numbers of factors and observations, we maximize the D-efficiency $D_e = 100 \times |\boldsymbol{X}'\boldsymbol{X}|^{1/(v+1)}/n$ of the design, which is a more convenient metric that ranges from 0 to 100 %. The use of the D-efficiency as a quality measure has an important drawback. Due to the fact that it is a scaled value (with respect to the numbers of factors and observations), its effectiveness to show a variation of a design's quality decreases with the size of the experiment. Even for completely random designs, the D-efficiency tends to be larger for experiments with large numbers of factors and observations. As a result, it is hard to obtain noticeable increases in the D-efficiency for large-scale experiments. This phenomenon can be observed in the computational experiments shown in Sect. 4.1.

In this paper, we focus on screening experiments involving main-effects linear regression models. This kind of experiments is widely used in industry, especially during the first stages of process analysis and the design phase of a new product (Montgomery and Jennings 2006). The main idea of screening is to carry out one relatively simple experiment with many factors in order to determine those that affect the response variable most. Once the influential factors have been isolated, a more complex experiment can be carried out in order to investigate these factors in more depth. For this reason, screening experiments usually consider main-effects regression models that only include first-order terms. Additionally, theoretical results show that optimal designs for screening experiments with quantitative factors only involve the extreme two levels of each factor (Mitchell 1974). In this sense, considering only two levels per factor (typically coded as -1 and $+1$) suffices to find an optimal design. Two-level full factorial, fractional factorial and Plackett and Burman (1946) designs are recommended in classical textbooks for this type of experiments (Atkinson et al. 2007; Montgomery 2008; Wu and Hamada 2000). However, as explained in Sect. 1, these designs are often infeasible in practice due to complications that arise in real-life experimentation. A more practical approach to design experiments, in the form of flexible design construction algorithms, is therefore needed.

3 Existing design construction algorithms

Existing algorithms for the generation of optimal experimental designs can be studied from the perspective of heuristic optimization. The algorithms aim to find a solution to the problem (the design of the experiment) that maximizes the objective function (the D-optimality criterion). More generally, they belong to the family of metaheuristics called *local search* algorithms. The algorithms in this family generate an initial solution (or take a solution as an input parameter) and iteratively explore neighboring solutions in the solution space, i.e. solutions that differ only slightly from the current solution, in order to find solutions with a better quality. The scheme applied in each iteration is the following: generate a *neighborhood set* of solutions by applying a local search operator to the current solution, move to one of the neighboring solutions according to a *selection strategy* and continue the exploration. The algorithms stop when a locally optimal solution is found or when a termination condition is satisfied.

The algorithms for the optimal design of experiments can be categorized according to two criteria. The first and major one is related to the granularity of the data structure the algorithm works with. The most traditional algorithms use entire points of the design as the units to be modified in order to generate the neighborhood set. These algorithms are known as *point-based* algorithms. A more recent approach uses a lower-level unit and modifies individual coordinates of the design points instead. These algorithms are known as *coordinate-based* algorithms. The second classification criterion is related to the strategy the algorithm uses to select the next neighboring solution to explore. Most of the algorithms use an intuitive approach: they iteratively select a neighbor that improves the value of the objective function; this strategy is known as *hill climbing*. Other algorithms use more com-

plex strategies in order to avoid getting stuck in a locally optimal solution. *Simulated annealing* (Kirkpatrick et al. 1983), for example, allows random transitions to worse neighboring solutions with a probability that is progressively reduced. *Tabu search* (Glover 1986), on the other hand, accepts non-improving moves when a locally optimal solution has been reached, but it prevents the return to previously visited solutions by using a memory structure.

3.1 Point-based algorithms

The point-based approach is the dominant one in the optimal design of experiments literature. A large number of algorithms in this category apply a hill climbing selection strategy; the most important ones are Fedorov's algorithm (Fedorov 1972), the modified Fedorov algorithm (Cook and Nachtsheim 1980), the K-exchange algorithm (Johnson and Nachtsheim 1983) and the KL-exchange algorithm (Atkinson and Donev 1989). These algorithms share an important feature, namely the use of a set (or pool) of candidate design points. This set remains fixed during the execution of the algorithms and is used to generate the neighborhood set in each iteration. Consequently, the portion of the design space that is explored is limited by the size of the set and the elements it contains. This issue becomes more important when the numbers of factors and observations increase, and the construction of a candidate set might be even intractable. Additionally, the algorithms also share the same neighborhood structure. In each iteration, they generate the neighborhood set with solutions resulting from an exchange of one point in the design matrix for one point in the candidate set. Due to this characteristic, these algorithms are called *point-exchange algorithms*.

There are two key aspects that differentiate the algorithms: the size of the neighborhood set and the strategy to select the improved neighboring solution to continue the exploration. Fedorov's algorithm uses the neighborhood set obtained considering all possible point exchanges, and then selects the solution within that set that has the highest value of the objective function. This strategy is known as *best-improvement*. The modified Fedorov algorithm operates in a similar fashion, but it attempts to reduce the execution time by making all beneficial exchanges as soon as they are discovered. If a neighboring solution with a better objective function value is found during the generation of the neighborhood set, the algorithm immediately moves to that better solution. This strategy is known as *first-improvement*. The KL-exchange algorithm attempts to reduce the execution time by limiting the size of the neighborhood set. This algorithm only considers for exchange the K points of the design with the lowest prediction variance and the L points of the candidate set with the highest prediction variance. For a more detailed review of the previous algorithms, see Nguyen and Miller (1992).

Other point-based algorithms are adaptations of metaheuristics that use more complex selection strategies. The algorithms proposed by Haines (1987), Meyer and Nachtsheim (1988), and Lejeune (2003) are implementations of simulated annealing. The DETMAX algorithm proposed by Mitchell (1974) was one of the first algorithms to include a primitive notion of a memory structure, well before the tabu search was introduced in the literature by Glover (1986). Only many years later, Sung Jung and Jin Yum (1996) proposed a point-based algorithm for optimal design of experiments that follows Glover's tabu search principles.

3.2 Coordinate-based algorithms

The coordinate-based approach emerged with the CEA proposed by Meyer and Nachtsheim (1995). This algorithm employs the exchange of individual elements of the design matrix as the operation to generate the neighborhood set of solutions. A step-by-step illustration of the CEA can be found in Goos and Jones (2011). The columnwise-pairwise algorithm proposed by Li and Wu (1997) for the construction of supersaturated designs extended this idea by exchanging pairs of coordinates corresponding to the same factor. The objective of this extension is to maintain the balance property of the design during the search process. This means that each level of any given factor is used an equal number of times. The coordinate-based algorithms overcome the limitations of point-exchange algorithms regarding the use of a pool of candidate design points. Since these algorithms do not use candidate sets for the generation of the neighborhood set, the portion of the solution they explore is not restricted in advance and they are a viable option to design large-scale experiments. Every coordinate-based algorithm in the literature applies a hill climbing selection strategy. The implementation of more complex local search metaheuristics that use the coordinate-based approach remains unexplored. One of the main goals of this paper is to set the first steps in that direction.

3.3 Genetic algorithms

In recent years, there have been several attempts to develop genetic algorithms for the construction of experimental designs. A genetic algorithm is a population-based metaheuristic that mimics the process of natural evolution. Since genetic algorithms do not belong to the group of local search metaheuristics, we did not discuss them in the previous sections. One of the first genetic algorithms, introduced by Montepiedra et al. (1998), proposes a binary string for the encoding of the solutions. In contrast, the algorithms proposed by Borkowski (2003) and Heredia-Langner et al. (2003) use a decimal representation. Although these algorithms have been able to find designs comparable to the ones obtained by classi-

cal algorithms, their execution times are considerably longer. Therefore, they are not a viable alternative for the design of large-scale experiments. For more information about genetic algorithms and their application to other optimization problems, see Davis (1991).

4 Selection strategy of the coordinate-exchange algorithm

The selection strategy is one of the most important aspects in the structure of the CEA. It defines the order in which the coordinates of the design points are considered for exchange, or from a heuristic point of view, the order in which the neighboring solutions are calculated and visited. It has an important impact both on the execution time and the solution quality. The best-improvement strategy is to non-experts intuitively more appealing than any other strategy. One would expect to find a better locally optimal solution if the neighboring solution selected in each iteration is the one with the best objective function value. However, this is the most time consuming strategy and the first-improvement strategy appears to be a good trade-off between solution quality and execution time. In order to identify which is the best selection strategy for the CEA, it is necessary to carry out a deeper analysis of the performance of the algorithm. In this section, we first analyze the original selection strategy used by Meyer and Nachtsheim (1995). We also propose a new selection strategy based on an orthogonality measure of the design. Finally, we perform a thorough comparison of several selection strategies to identify the one that yields the best algorithm performance.

4.1 Original selection strategy

The selection strategy originally proposed for the CEA is derived from the point selection strategy of the K-exchange algorithm. This strategy takes into account the prediction variance at the design points to determine the subset of points considered for exchange. The prediction variance at a design point is defined as $\text{var}(\boldsymbol{x_i}) = \boldsymbol{f}'(\boldsymbol{x_i})(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{f}(\boldsymbol{x_i})$. The use of this value to guide the execution of the algorithm is motivated by the update formulas commonly used in point-based algorithms (Atkinson et al. 2007). These formulas allow a fast calculation of the determinant of the information matrix when a point is added to or deleted from the design. When a point $\boldsymbol{x_a}$ is added, the new value of the determinant can be calculated using the expression $|\boldsymbol{X}'\boldsymbol{X}|_{n+1} = |\boldsymbol{X}'\boldsymbol{X}|_n(1 + \text{var}(\boldsymbol{x_a}))$. Similarly, when a point $\boldsymbol{x_d}$ is deleted, the determinant value can be calculated using the expression $|\boldsymbol{X}'\boldsymbol{X}|_{n-1} = |\boldsymbol{X}'\boldsymbol{X}|_n(1 - \text{var}(\boldsymbol{x_d}))$. The point selection strategy of the K-exchange algorithm focuses on the K least critical points in the design, i.e. the points the deletion of which produces the smallest decrease in the determinant value. Therefore, the K-exchange algorithm only considers the K design points with the smallest prediction variance. The original CEA uses the same guideline in order to select the subset of design points to be modified in each iteration. The size of this subset is equal to $K = n/4$, which is suggested to be sufficient in order to obtain designs comparable to those produced by the modified Fedorov algorithm (Meyer and Nachtsheim 1995).

The original computational experiments carried out to measure the performance of the CEA do not include a formal study regarding the effects of the variation of the K value. Additionally, only main-effects models with up to 11 factors were considered. It is important to determine the appropriateness of the value $K = n/4$, especially for the design of large-scale experiments. For this reason, we carried out additional computational experiments. For this study and the others presented in this paper, the benchmark set is composed of 28 instances with the number of factors ranging from 3 to 30. For practical purposes, we divide the benchmark set into small experiments (with less than 11 factors), medium-sized experiments (from 11 to 20 factors) and large experiments (from 21 to 30 factors). The entire set of benchmark instances is detailed in Table 1. Every experiment considered in this set has a number of observations that is a multiple of four. Whenever n is a multiple of four, there exist orthogonal designs for which $|\boldsymbol{X}'\boldsymbol{X}| = n^{v+1}$. Therefore, the D-efficiency of the optimal designs is known to be 100 % for all the experiments in the benchmark set. This explains why we chose experiments with these characteristics: in order to have certain information about the D-efficiency of their optimal designs. However, the algorithms analysed and proposed in this paper are meant to generate designs for experiments with any numbers of factors and observations, possibly with constraints on the factor levels. In order to explore a variety of experimental scenarios, the numbers of observations were calculated by considering four different values for the ratio n/v. There are seven experiments in the benchmark set for each ratio in the set $\{1, 1.5, 2, 3\}$. Whenever needed, the actual numbers of observations were calculated by rounding up to the next higher multiple of four. The metric used to measure the computational effort of the algorithm is the number of evaluations of the determinant $|\boldsymbol{X}'\boldsymbol{X}|$. We decided not to use the execution time because it is severely affected by the use of update formulas, which is outside the scope of this paper.

We tested the effect of four different values of K on the performance of the CEA. The algorithm was executed 1,000 times for each instance in the benchmark set, each time starting from a different random initial design. Six quality measures were computed for each set of runs: the minimum, the average and the maximum D-efficiency of the designs, and the minimum, the average and the maximum number of determinant evaluations. The averages of these measures for

Table 1 Benchmark set of experiments

Small		Medium-sized		Large	
v	n	v	n	v	n
3	4	11	12	21	44
4	8	12	20	22	68
5	12	13	28	23	24
6	20	14	44	24	36
7	8	15	16	25	52
8	12	16	24	26	80
9	20	17	36	27	28
10	32	18	56	28	44
		19	20	29	60
		20	32	30	92

Table 2 Effect of the K value on the average design quality and the average number of determinant evaluations of the CEA

K value	D-efficiency			Det. evaluations		
	Min.	Avg.	Max.	Min.	Avg.	Max.
Small experiments						
$n/4$	85.26	93.44	99.93	70	150	290
$n/3$	83.11	93.46	99.93	87	322	169
$n/2$	87.61	93.72	99.96	138	228	474
n	88.43	94.95	99.95	214	350	640
Medium-sized experiments						
$n/4$	88.58	94.51	97.55	543	1,271	2,606
$n/3$	90.28	94.95	98.12	658	1,498	3,041
$n/2$	90.92	95.49	98.08	956	2,016	4,341
n	92.97	96.26	98.73	1,450	3,033	6,244
Large experiments						
$n/4$	94.73	96.52	97.34	2,412	5,557	11,958
$n/3$	95.16	96.71	97.46	2,934	6,665	13,756
$n/2$	95.78	96.91	97.56	4,172	8,788	19,113
n	96.52	97.18	97.72	6,038	13,421	27,646

each subset of benchmark experiments are shown in Table 2. First of all, observe that the variability of the D-efficiency decreases with the size of the experiment. The D-efficiencies of the designs for small experiments lie within the interval [85, 100], while for large experiments the D-efficiencies lie within the interval [94, 98]. This illustrates the drawback of the D-efficiency explained in Sect. 2. Secondly, observe that the K value has a considerable impact on the quality of the designs obtained and on the execution time of the algorithm. For small designs, the average maximum D-efficiency does not substantially change with the K value, even though the minimum and the average D-efficiency increase with K. Hence, small values of K do not negatively affect the effectiveness of the algorithm when it is executedseveral times starting from different initial designs. However, the trend is different for medium-sized and large experiments. The K value has a substantial impact on all D-efficiency measures for these experiments. It appears that the trade-off between the execution time of the algorithm and the quality of the designs obtained is well defined. Small values of K reduce the execution time of the algorithm but they also reduce its capacity to find better designs. Additionally, observe that the CEA cannot find the optimal designs for medium-sized and large experiments, not even when $K = n$ and the algorithm explores all the design points. This shows the limited capability of the CEA for generating optimal designs for experiments with large numbers of factors and observations.

4.2 Orthogonality-based selection strategy

An orthogonal design is characterized by a 100 % D-efficiency and is achieved when all the columns of the design matrix are orthogonal. The off-diagonal elements of the information matrix are equal to zero; consequently, the estimates of the parameters of the regression model are not correlated and $|\boldsymbol{X}'\boldsymbol{X}| = n^{v+1}$. Any measure of how orthogonal the columns are is therefore a good indicator of the quality of the design. In this section, we propose a new selection strategy that uses an orthogonality measure of the design in order to guide the execution of the CEA. The main objective is to start the exploration of the design matrix in an area where an exchange is likely to produce a large improvement in the D-efficiency, and avoid the exploration of poor exchanges.

For every pair of columns i and j of the design matrix, such that $0 \leq i \leq v$ and $0 \leq j \leq v$, the inner product between the two columns is given by the elements $(\boldsymbol{X}'\boldsymbol{X})_{i,j}$ and $(\boldsymbol{X}'\boldsymbol{X})_{j,i}$. A global non-orthogonality measure of a column k relative to the others can be defined as $\theta_k = \sum_{m=0}^{v} (\boldsymbol{X}'\boldsymbol{X})_{m,k}^2$. The larger the θ_k value, the less orthogonal the column k is to the others and therefore the larger its potential for improvement. The use of the squared value instead of the absolute value results in a severe penalization for columns that are highly non-orthogonal. The inclusion of the element $(\boldsymbol{X}'\boldsymbol{X})_{0,k}$ in the definition of θ_k penalizes the unbalanced columns. This is useful because orthogonal designs are known to be balanced. The proposed orthogonality measure is very similar to the $E(s^2)$ optimality criterion proposed by Booth and Cox (1962) for the generation of supersaturated designs. It has been shown that the minimization of $\sum_{k=0}^{v} \theta_k$ is an approximation to the optimization of the A-optimality and the D-optimality criteria (Nguyen 1996).

In each execution, our modified version of the CEA sorts the columns of the design matrix in decreasing order of θ_k. The CEA then explores the design matrix column by column in that order. If an exchange that produced an improvement in the determinant $|\boldsymbol{X}'\boldsymbol{X}|$ is found when the exploration of a column has been completed, the current execution is stopped and a new execution is started. The pseudocode of the CEA with the orthogonality-based selection strategy is shown in Algorithm 1. Note that in the implementation for the design of screening experiments, which is the topic of this paper, the exchange procedure only exchanges the sign of the coordinate (from -1 to $+1$, or vice versa).

Algorithm 1: CEA with the orthogonality-based selection strategy

Input: Design matrix $\boldsymbol{X}$

```
for j ← 1 to v do
    θ_j ← Σ_{i=0}^{v} (X'X)_{i,j}^2
Define (c_1, c_2, ..., c_v) as a permutation of integers (1, 2, ..., v)
such that θ_{c_1} ≥ θ_{c_2} ≥ ··· ≥ θ_{c_v}
d_0 ← |X'X|
d ← d_0
for j ← 1 to v do
    for i ← 1 to n do
        exchange(x_{ic_j})
        if |X'X| > d then
            d ← |X'X|
        else
            revert_exchange(x_{ic_j})
    if d > d_0 then
        return CEA(X)
```

4.3 Comparison of selection strategies

There exist alternative selection strategies for the CEA the performances of which are not well documented in the literature. The first alternative is a best-improvement selection strategy. Also other simpler first-improvement selection strategies might avoid the calculation of the prediction variance at each design point, and therefore reduce the computation time. In this section, we report the results of a computational experiment carried out in order to determine the best selection strategy for the design of screening experiments in terms of design quality and execution time. We tested six different strategies in the way described in Sect. 4.1:

- *Row-based*: explores the elements of the design matrix row by row.
- *Column-based*: explores the elements of the design matrix column by column.
- *Variance-based*: explores the design points in increasing order of their prediction variance.
- *Orthogonality-based:* explores the columns of the design matrix in decreasing order of their global non-orthogonality measure θ_k.
- *Best-improvement*: explores all the elements of the design matrix and exchanges the one that produces the largest improvement to the objective function.
- *Mildest improvement*: explores all the elements of the design matrix and exchanges the one that produces the smallest improvement to the objective function.

The first four strategies are first-improvement strategies and have linear order of time complexity with respect to the number of elements in the design matrix. The other two strategies have quadratic order of time complexity. The results of the computational experiment are shown in Table 3. We stick to the use of the number of determinant evaluations as a performance metric because it provides a good approximation to the complete execution time. By doing so, we focus on the performance differences caused by the selec-

Table 3 Overall effect of the selection strategy on the average design quality and the average number of determinant evaluations of the CEA

Strategy	D-efficiency			Det. evaluations		
	Min.	Avg.	Max.	Min.	Avg.	Max.
Linear						
Row	93.05	96.23	98.75	2,866	6,019	12,590
Column	93.44	96.18	98.72	3,045	6,188	12,820
Variance	93.30	96.18	98.75	2,804	6,008	12,711
Orthogonality	93.95	96.70	98.75	2,105	4,454	9,137
Quadratic						
Best-imp.	93.25	96.38	98.66	77,550	97,827	121,594
Mildest-imp.	93.40	96.38	98.68	78,854	98,517	122,281

Table 4 Breakdown of the effect of the selection strategy on the average design quality and the average number of determinant evaluations of the CEA for small, medium-sized and large experiments

Strategy	D-efficiency			Det. evaluations		
	Min.	Avg.	Max.	Min.	Avg.	Max.
Small experiments						
Row	88.94	95.03	99.96	214	366	704
Variance	89.60	94.84	99.91	214	350	592
Orthogonality	90.71	96.43	99.96	118	228	408
Medium-sized experiments						
Row	92.97	96.24	98.85	1,386	3,052	6,342
Variance	92.99	96.24	98.85	1,550	3,032	6,568
Orthogonality	93.98	96.37	98.80	1,047	2,051	4,166
Large experiments						
Row	96.43	97.19	97.68	6,468	13,509	28,345
Variance	96.57	97.18	97.69	6,130	13,374	28,550
Orthogonality	96.53	97.20	97.70	4,754	10,239	21,092

tion strategy and not by the updated formulas used or any other implementation detail. Although the execution times of certain intermediate steps in some selection strategies are ignored in this approach (for example, the calculation of the prediction variance at each design point in the variance-based selection strategy, and the sorting of the columns of the design matrix according to the orthogonality measure in the orthogonality-based selection strategy), this has no meaningful impact on the results. Consequently, an increment of the number of determinant evaluations can be safely translated into an increment of the execution time.

It can be observed that, despite the fact that the selection strategies with quadratic time complexity require much longer execution times, the quality of the designs obtained does not substantially differ from the ones obtained by the selection strategies with linear time complexity. This shows that the extra computational effort is in vain. Secondly, the variance-based strategy has a similar performance to the row-based and column-based strategies, both in terms of D-efficiency and number of determinant evaluations. This shows that the use of the prediction variance as a mechanism to guide the CEA does not provide any added value. Finally, the orthogonality-based strategy was able to generate slightly better designs (on average) than the other strategies using a 30 % shorter execution time.

Table 4 shows a breakdown of the results in Table 3 for the best selection strategies. It can be observed that the orthogonality-based strategy on average finds better designs for small experiments. For medium-sized and large experiments, the three strategies produce designs of very similar quality. However, the orthogonality-based selection strategy requires a substantially shorter execution time.

The orthogonality-based selection strategy clearly results in the best performance of the CEA. The use of the prediction variance to guide the CEA does not provide any added value in comparison to simpler strategies, even though the variance-based selection strategy has often been used in the literature as an alternative to first-improvement and best-improvement

strategies. The fact that the three strategies lead to designs of similar quality for large-scale experiments suggests that the CEA cannot be improved by modifying the selection strategy.

5 Iterated local search algorithm

One of the weakest aspects of the CEA is its tendency to get trapped in locally optimal designs. To overcome this issue, it is usual to execute the algorithm several times, each time starting from a different randomly generated design. In this section, we propose an ILS algorithm that uses a more selective strategy than the previous random sampling of initial solutions. The main premise of this algorithm is that better solutions are probably found in the space that surrounds the current locally optimal solution, but not close enough to be reachable by the neighborhood structure of the local search. The more selective strategy is also motivated by the observation that a locally optimal solution is generally already a quite good solution, so that it is sensible to retain at least some part of that structure when it is modified. Therefore, in contrast to completely disregarding the locally optimal solution, as it is done by multi-start algorithms, the ILS performs a small modification to the locally optimal solution called perturbation. The slightly modified solution is then used as an initial solution for the next iteration of the local search. The size of the perturbation is a key factor of the algorithm because it determines what portion of the locally optimal solution is altered, i.e. how considerable the modification is. If the perturbation is too large, important information contained within the current locally optimal solution is lost and the algorithm behaves as if a random restart were used. On the other hand, if the perturbation is too small, the algorithm is not able to escape from the current local optimum and the iterative process will not add any value. The general pseudocode of the ILS is shown in Algorithm 2.

Algorithm 2: Iterated local search algorithm

Input: An initial solution S_0

$S_{best} \longleftarrow \text{LocalSearch}(S_0)$
while $\neg$ *TerminationCondition* **do**
  $S_{pert} \longleftarrow \text{Perturb}(S_{best})$
  $S_{pert} \leftarrow \text{LocalSearch}(S_{pert})$
  if *AcceptanceCriterion*(S_{best}, S_{pert}) **then**
    $S_{best} \leftarrow S_{pert}$
return S_{best}

In order to implement an ILS algorithm for the optimal design of experiments, it is necessary to define the mechanism used to generate the initial design, the local search algorithm, the perturbation operator, the termination condition of the algorithm and the acceptance criterion used to update the current design. In the algorithm presented in this paper, the local search procedure is the CEA with the orthogonality-based selection strategy. The termination condition stops the execution of the algorithm when a fixed number of iterations has been performed without finding a better design. The acceptance criterion updates the current design only when a new better design has been found. The two remaining building blocks are explained in the following sections.

5.1 Initial design

We developed a new greedy algorithm for the construction of initial designs. The algorithm iteratively adds points in such a way that the columns of the design matrix remain as orthogonal as possible. This approach is similar in spirit to that proposed by Mandal and Koukouvinos (2014) for the generation of optimal multi-level supersaturated designs using integer programming.

The first point of the design is randomly chosen, which enables the algorithm to obtain different initial designs for the same experiment. The algorithm then generates the remaining points one by one, coordinate by coordinate, considering the orthogonality of the partial design built so far. At each step, the algorithm identifies the two columns that are least orthogonal, i.e. the ones with the maximum absolute inner product. The coordinates corresponding to these two columns are assigned values that reduce the inner product and make the columns more orthogonal. Each of the remaining columns are sorted in decreasing order of the non-orthogonality measure θ_k. The resulting column ranking defines the order in which the coordinates are assigned values. The value chosen for each coordinate is the one that minimizes θ_k.

The following notation is used to explain the initial design construction algorithm. Let $\boldsymbol{X}_{1:k,i}$ be a column vector that contains the first k elements of the i-th column of the partial design matrix. Let $\boldsymbol{X}_{1:k,m} \cdot \boldsymbol{X}_{1:k,n}$ be the inner product of the m-th and the n-th column vectors. Finally, let $\theta_{l[k]}$ be the global non-orthogonality measure of the l-th column $\boldsymbol{X}_{1:k,l}$. The pseudocode of the initial design construction algorithm is shown in Algorithm 3.

In contrast to other algorithms for the generation of initial designs available in the literature (Atkinson and Donev 1989; Lejeune 2003), the proposed algorithm involves a pure coordinate-based approach. It has a polynomial time complexity $\Theta(nv^2)$, which makes it perfectly useful for the generation of designs for large-scale experiments. We compared the performance of the newly proposed greedy algorithm to that of a procedure that uses random initial designs. The experiment was executed in the same way as the experiments described in previous sections. The results are shown in Table 5. The execution times are not shown because they

Algorithm 3: Initial design construction algorithm

Output: Initial design matrix $\boldsymbol{X}$

for $i \leftarrow 1$ **to** v **do**
  $x_{1i} \leftarrow$ Random level
for $k \leftarrow 2$ **to** n **do**
  Find integers $a, b \in \{1, 2, \ldots, v\}$ such $a \neq b$ and $|X_{1:k-1,a} \cdot X_{1:k-1,b}|$ is maximum
  $x_{ka}, x_{kb} \leftarrow$ Levels that minimize $|X_{1:k,a} \cdot X_{1:k,b}|$
  Define $(c_1, c_2, \ldots, c_{v-2})$ as a permutation of integers $\in \{1, 2, \ldots, v\} \setminus \{a, b\}$ such that $\theta_{c_1[k-1]} \geq \theta_{c_2[k-1]} \geq \cdots \geq \theta_{c_{v-2}[k-1]}$
  for $i \leftarrow 1$ **to** $v - 2$ **do**
    $x_{kc_i} \leftarrow$ Level that minimizes $\theta_{c_i[k]}$

Table 5 Comparison of the D-efficiency of the designs obtained by the greedy construction algorithm and a random procedure

Algorithm	Min.	Avg.	Max.
Random	55.25	66.38	78.80
Greedy	85.52	93.84	97.03

are negligible. It can be observed that the greedy algorithm obtains designs with a much higher quality.

5.2 Perturbation operator

The operator used to perturb the locally optimal designs exchanges coordinates of the design matrix. We developed two versions of the operator. The first version randomly selects the coordinates that are going to be exchanged. The second version modifies columns of the design that have large values of θ_k with higher probability. In the second approach, a column i is randomly selected, and it is modified with a probability $\theta_i / \max(\theta_k)$, for $1 \leq k \leq v$. If the selected column is modified, the coordinate to be exchanged is randomly selected. If the selected column is not modified, another column is randomly selected and evaluated for a possible modification.

The size of the perturbation is defined by the number of coordinates that are exchanged. This number is selected randomly in each execution from a discrete uniform distribution on the set $\{1, 2, \ldots, \lambda\}$. The λ value is the upper bound of the number of coordinates that can be exchanged and is the only parameter that defines the size of the perturbation operator. Additionally, two mechanisms for the adjustment of the perturbation size, given by λ, were studied. The first one is a simple static approach: the size of the perturbation stays fixed during the entire execution of the algorithm. The second one is a reactive approach: the algorithm dynamically modifies λ according to the results of the exploration. Every time the algorithm finds a better design, the size of the perturbation is set to the minimum value of 1 in order to explore the immediate neighborhood of the design. If no better design is found during the next few iterations, λ is progressively increased in order to augment the probability of escaping from the locally optimal design. The type of the perturbation operator, the size λ of the perturbation and the mechanism for its adjustment are input parameters of the algorithm.

6 Parameter tuning and statistical analysis

In this section, we describe the study carried out to identify the parameters that impact the execution of the ILS algorithm and to determine the combination of parameters that yields the best performance. The following parameters were studied using a full factorial experiment:

- The number of iterations the algorithm is allowed to execute without finding a better design (`num_it`).
- The size λ of the perturbation operator (`pert_size`).
- The mechanism for the adjustment of the perturbation size (`pert_adj`), that can be either static (`stat`) or reactive (`react`).
- The perturbation operator (`pert_opt`), that can be either random (`rand`) or orthogonality-based (`orth`).
- The number of restarts of the algorithm (`num_rest`).

The different levels tested for each parameter are shown in Table 6. Note that the perturbation size is expressed in percentages of the number of coordinates in the design matrix ($v \times n$). Each configuration was executed once for each instance of the benchmark set, resulting in a total of $2^2 \times 3^2 \times 5 \times 28 = 5{,}040$ observations. For each observation, the D-efficiency of the design obtained and the number of determinant evaluations the algorithm performed were determined. For each of these measures, a mixed-effects analysis of variance (ANOVA) was conducted using the statistical software JMP. Both models use a random effect for each instance of the benchmark set to indicate that all the measurements for the same instance are correlated. The p values of the F-tests that determine the significance of each parameter and interaction in the ANOVA models are shown in Table 7. A bold p value indicates that the parameter or interaction has a significant impact.

The parameters `num_rest` and `num_it` have a significant impact on both measures. These parameters mainly define the execution time of the algorithm and it is reasonable to think that the larger their values, the better the quality of the designs obtained. It is also reasonable to expect `pert_size` to have a significant impact. The larger the perturbation, the larger the portion of the solution solution space explored, but also the larger the time that is required by the CEA to improve the modified design.

Table 6 Parameters and levels tested

Parameter	Levels
num_rest	1, 10, 25
num_it	100, 1,000, 5,000
pert_size	1, 5, 10, 15, 20 %
pert_adj	stat, react
pert_opt	rand, orth

Table 7 p values of the F-tests to determine the significance of each term in the ANOVA models

Factor/interaction	D-efficiency	Det. evaluations
num_rest	< **0.0001**	< **0.0001**
num_it	< **0.0001**	< **0.0001**
pert_size	< **0.0001**	< **0.0001**
pert_adj	0.0754	< **0.0001**
pert_opt	**0.0130**	0.6759
num_rest × num_it	**0.0423**	< **0.0001**
num_rest × pert_size	< **0.0001**	< **0.0001**
num_rest × pert_adj	0.1666	< **0.0001**
num_rest × pert_opt	0.8495	0.7599
num_it × pert_size	**0.0204**	< **0.0001**
num_it × pert_adj	0.5597	< **0.0001**
num_it × pert_opt	0.6119	0.6345
pert_size × pert_adj	0.0606	**0.0013**
pert_size × pert_opt	0.4214	0.9991
pert_adj × pert_opt	0.7596	0.7111

One of the most important aspects to be determined with the statistical analysis is the effectiveness of both diversification mechanisms, the number of restarts (num_rest) and the size of the perturbation operator (pert_size), and the interaction between them. Both of these parameters and their interactions are highly significant in both ANOVA models. The average D-efficiency and the average number of determinant evaluations for each combination of levels for both parameters are shown in Fig. 1. It can be observed that, even though the application of the perturbation operator considerably improves the overall performance of the algorithm, the use of the restart mechanism is still necessary. Nevertheless, when num_rest exceeds 10, the increase in the quality of the designs obtained is not substantial. Additionally, the size of the perturbation operator that leads to the best algorithm performance is pert_size = 10 %. Larger values of pert_size only increase the execution time of the algorithm and not the quality of the designs obtained.

Figure 2 shows the average D-efficiency and the average number of determinant evaluations for each level of the parameter num_it. This parameter is highly significant in both ANOVA models and appears in two interactions that are bor-

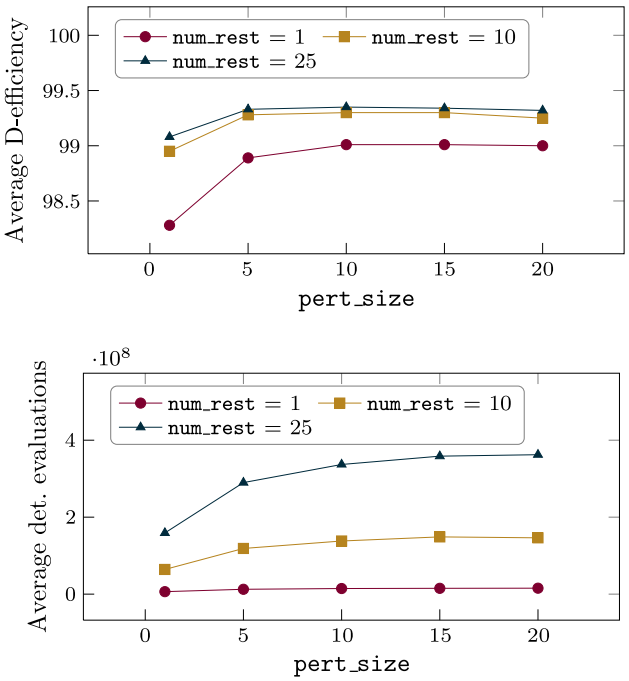

Fig. 1 Influence of the interaction num_rest × pert_size on the design quality and the number of determinant evaluations

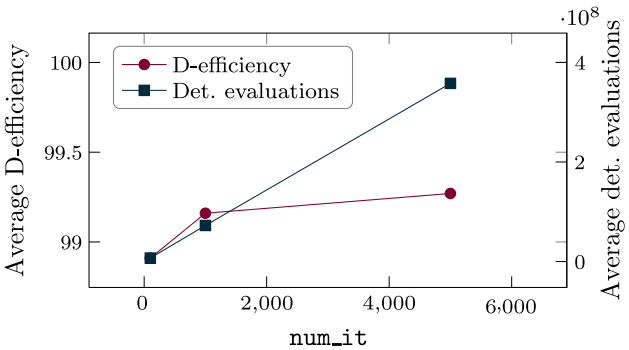

Fig. 2 Influence of the parameter num_it on the design quality and the number of determinant evaluations

derline significant. It can be observed that the increase in the design quality attenuates when the number of iterations exceeds 1,000, while the number of determinant evaluations increases linearly.

Figures 3 and 4 show the average D-efficiency and the average number of determinant evaluations for each level of the parameters pert_adj and pert_opt, respectively. It can be observed that the reactive mechanism for the adjustment of the perturbation size (react) results in designs with a quality that is comparable to the quality of the ones obtained by the static mechanism (stat). However, it reduces the execution time of the algorithm by half (p value < 0.0001). This shows the great importance of the reactive mechanism for the reduction of the execution time of the algorithm. The orthogonality-based version of the perturbation

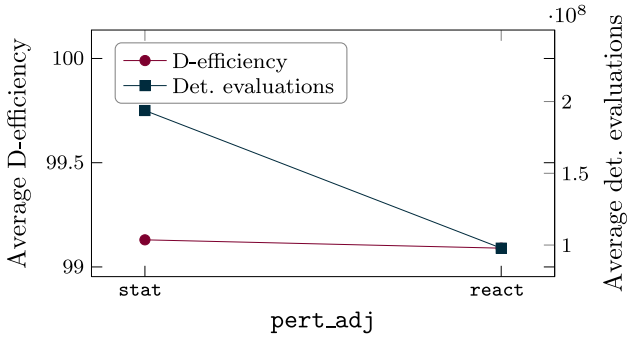

Fig. 3 Influence of the parameter `pert_adj` on the design quality and the number of determinant evaluations

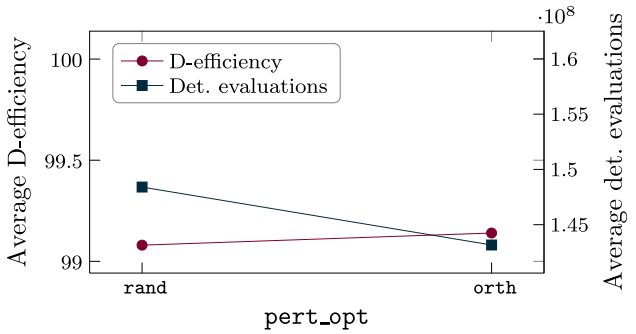

Fig. 4 Influence of the parameter `pert_opt` on the design quality and the number of determinant evaluations

operator (`orth`) obtains designs with slightly better quality (p value $= 0.0130$) using an execution time comparable to the one used by the random version (`rand`).

The analysis of variance suggests using the following settings for the parameters of our algorithm: `num_rest` = 10, `pert_size` = 10 %, `pert_adj` = `react` and `pert_opt` = `orth`. The number of iterations (`num_it`) is the parameter by means of which the user can specify the desired execution time of the algorithm.

7 ILS versus the CEA implemented in commercial software

In this section, we compare the performance of the ILS algorithm to the implementation of the CEA in the statistical package JMP and described in Chapter 2 of Goos and Jones (2011). Unlike the original CEA proposed by Meyer and Nachtsheim, the CEA implemented in JMP uses a row-based selection strategy. Since the JMP software does not report the number of determinant evaluations performed for the construction of the design, we rely on the execution time as the performance metric to measure the computational effort. The comparison using this metric is not completely fair because JMP, being a specialized statistical software, implements update formulas for matrix determinants in order to reduce

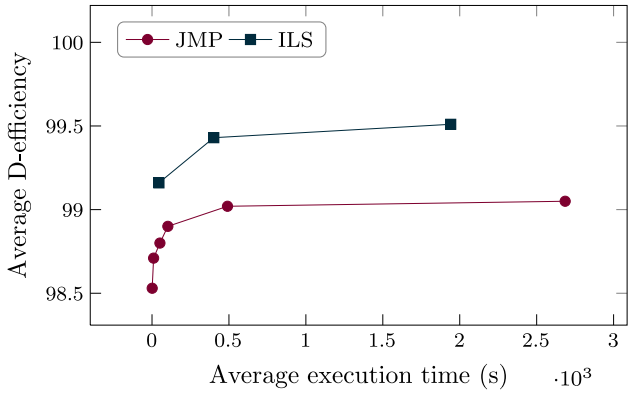

Fig. 5 Performance comparison between the ILS algorithm and the CEA found in the statistical software package JMP for all experiments in the benchmark set

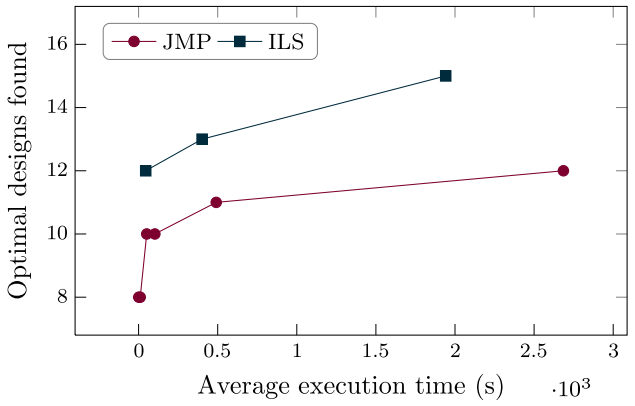

Fig. 6 Number of optimal designs found by the ILS algorithm and the CEA found in the statistical software package JMP

the execution time of the algorithm (see, for instance, Atkinson et al. 2007; Meyer and Nachtsheim 1995). Hence, our comparison favors the JMP implementation of the CEA.

The number of restarts is the only parameter that can be specified in JMP's interface in order to define the computational effort of the algorithm. We compute designs for the benchmark set of experiments using 100, 1,000, 5,000, 10,000, 50,000 and 250,000 restarts, and compare the results with the ones obtained by the ILS using the parameter configuration identified in Sect. 6. The numbers of iterations specified for the ILS algorithm were the ones shown in Table 6. The D-efficiency and execution time, averaged over all instances in the benchmark set, are shown for each algorithm in Fig. 5.

It is clear that the proposed ILS algorithm outperforms the CEA found in JMP. Even with very small computational effort, the ILS finds designs with a higher D-efficiency than does the longest execution of the CEA algorithm in JMP. Figure 6 shows the number of optimal designs found by each algorithm for the experiments in the benchmark set. The ILS algorithm is able to find three optimal designs more than the CEA in JMP. These designs correspond to the experiment

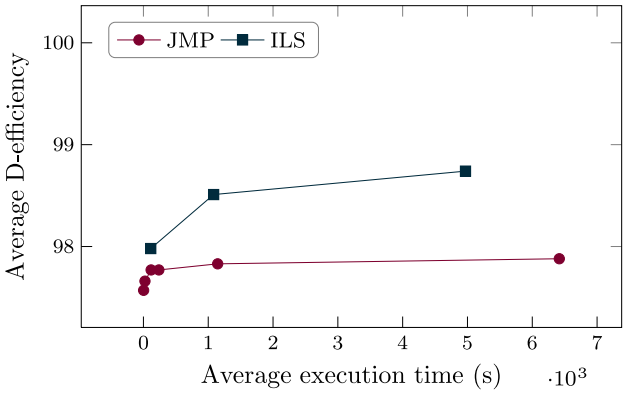

Fig. 7 Performance comparison between the ILS algorithm and the CEA found in the statistical software package JMP for large experiments in the benchmark set

with 13 factors and 28 observations, the one with 14 factors and 44 observations and that with 16 factors and 24 observations. The CEA in JMP is able to find 12 optimal designs while performing the most demanding execution (with the longest execution time). The ILS algorithm is, however, able to match this number of optimal designs found while performing the execution with the shortest execution time.

Figure 7 shows the average D-efficiency and execution time of each algorithm for the design of large experiments only. Observe that the ILS algorithm is able to produce designs with an average D-efficiency that is almost 1 % higher than that of the designs produced by the CEA found in JMP. Considering the properties of the D-efficiency when used as a quality measure for large experimental designs (described in Sect. 2 and shown in Sect. 4.1), this 1 % improvement is substantial. Additionally, observe that, when increasing the execution time of the algorithms, the improvement in the average D-efficiency achieved by the ILS algorithm is considerably larger than that achieved by the CEA found in JMP. The speed of the ILS algorithm could even be improved by developing new update formulas similar to those proposed in Arnouts and Goos (2010), Goos and Vandebroek (2003) and implemented in JMP. We leave this for future research, as this paper concentrates on strategies for making the CEA more effective.

8 Conclusions

In this paper, we have dealt with the construction of D-optimal experimental designs for main-effects models. An analysis of the coordinate-exchange algorithm led to two important conclusions. First, restricting the number of design points considered for replacement in each iteration of the algorithm, usually labelled K in the literature, has a significant negative impact on the quality of the designs obtained. In other words, the exploration of a small subset of design points reduces the execution time of the algorithm, but it also reduces the capability of finding good designs. Secondly, even though most of the available algorithms in the literature (both point-based and coordinate-based) use the prediction variance at the design points to guide the optimization, we provide strong evidence that this strategy does not add any value to the performance of the algorithm. Instead, we propose a new selection strategy for the algorithm based on the orthogonality of the columns of the design matrix. This strategy allows the construction of better designs for small experiments and reduces the execution time of the algorithm by about 30 % for medium-sized and large experiments.

We also propose a new iterated local search algorithm for the optimal design of experiments. The orthogonality-based strategy is included in two main components of the iterated local search: the greedy algorithm for the construction of initial designs and the perturbation operator. Additionally, the metaheuristic uses a reactive strategy to dynamically modify the size of the perturbation during the execution of the algorithm. We performed an extensive statistical analysis in order to identify the important components of the algorithm and to validate their effectiveness. The proposed algorithm outperforms the most important commercial statistical package for optimal design of experiments in terms of both execution time and design quality, especially for experiments with large numbers of factors and observations.

The iterated local search algorithm can also be applied for the generation of experimental designs that involve more general models (i.e. linear models that include interaction effects and/or quadratic effects, and even for non-linear models). This is because the algorithm only modifies the original design matrix, which is independent of the model expansion $\boldsymbol{f}'(\boldsymbol{x})$ of the design points. The orthogonality measure might also be useful to guide the coordinate-exchange algorithm while considering other optimality criteria. Orthogonality is a property that is desirable also in the context of A-, V- and G-optimality. Nevertheless, the implementation of the orthogonality-based selection strategy for more general models is not straightforward. It is necessary to develop a more general orthogonality measure that considers the particular structure of the extended design matrix in the model. One possible extension could be to consider all the columns of the design matrix that involve the factor being evaluated. By doing so, it would be possible to have a general measure of the degree of non-orthogonality presumably caused by the factor's levels. The interchange algorithm proposed by Trinca and Gilmour (2000) for blocking response surface designs guides its execution using a similar strategy. Nevertheless, it is important to point out that the D-efficiency gain achieved by the iterated local search algorithm (with respect to the regular coordinate-exchange algorithm) is not entirely due to the use of the orthogonality measure. The quality of the

designs obtained by the iterated local search algorithm only slightly decreases when an alternative selection strategy (not based on orthogonality) is used, but it remains better than that of the designs generated by the regular coordinate-exchange algorithm. We believe therefore that exploring the usefulness of our iterated local search algorithm for models other than main-effects models would be very useful.

Acknowledgments We acknowledge the financial support of the Flemish Fund for Scientific Research (FWO).

References

Arnouts, H., Goos, P.: Update formulas for split–plot and block designs. Comput. Stat. Data Anal. **54**(12), 3381–3391 (2010)

Atkinson, A.C., Donev, A.N.: The construction of exact D-optimum experimental designs with application to blocking response surface designs. Biometrika **76**(3), 515 (1989)

Atkinson, A.C., Donev, A.N., Tobias, R.: Optimum Experimental Designs, with SAS, vol. 34. Oxford University Press, Oxford (2007)

Booth, K.H.V., Cox, D.R.: Some systematic supersaturated designs. Technometrics **4**(4), 489–495 (1962)

Borkowski, J.J.: Using a genetic algorithm to generate small exact response surface designs. J. Probab. Stat. Sci. **1**(1), 65–88 (2003)

Cawse, J.N.: The combinatorial challenge. In: Cawse, J.N. (ed.) Experimental Design for Combinatorial and High Throughput Materials Development, pp. 1–26. Wiley, Hoboken, NJ (2003)

Cook, R.D., Nachtsheim, C.J.: A comparison of algorithms for constructing exact D-optimal designs. Technometrics **22**(3), 315–324 (1980)

Davis, L. (ed.): Handbook of Genetic Algorithms, vol. 115. Van Nostrand Reinhold, New York (1991)

Fedorov, V.V.: Theory of Optimal Experiments. Academic Press, New York (1972)

Glover, F.W.: Future paths for integer programming and links to artificial intelligence. Comput. Oper. Res. **13**(5), 533–549 (1986)

Goos, P., Vandebroek, M.: D-optimal split–plot designs with given numbers and sizes of whole plots. Technometrics **45**(3), 235–245 (2003)

Goos, P., Jones, B.: Optimal Design of Experiments: A Case Study Approach. Wiley, New York (2011)

Haines, L.M.: The application of the annealing algorithm to the construction of exact optimal designs for linear-regression models. Technometrics **29**(4), 439–447 (1987)

Heredia-Langner, A., Carlyle, W.M., Montgomery, D.C., Borror, C.M., Runger, G.C.: Genetic algorithms for the construction of D-optimal designs. J. Qual. Technol. **35**, 28–46 (2003)

Johnson, M.E., Nachtsheim, C.J.: Some guidelines for constructing exact D-optimal designs on convex design spaces. Technometrics **25**(3), 271–277 (1983)

Jones, B.: Computer aided designs for practical experimentation. Ph.D. thesis, University of Antwerp, Faculty of Applied Economics (2008)

Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science **220**(4598), 671 (1983)

Lejeune, M.A.: Heuristic optimization of experimental designs. Eur. J. Oper. Res. **147**(3), 484–498 (2003)

Li, W.W., Wu, C.F.J.: Columnwise–pairwise algorithms with applications to the construction of supersaturated designs. Technometrics **39**(2), 171–179 (1997)

Lourenco, H.R., Martin, O.C., Stützle, T.: Iterated local search. In: Glover, F., Kochenberger, G.A. (eds.) Handbook of Metaheuristics, pp. 320–353. Springer, Berlin (2003)

Mandal, B.N., Koukouvinos, C.: Optimal multi-level supersaturated designs through integer programming. Stat. Probab. Lett. **84**, 183–191 (2014)

Meyer, R.K., Nachtsheim, C.J.: Simulated annealing in the construction of exact optimal design of experiments. Am. J. Math. Manag. Sci. **8**, 329–359 (1988)

Meyer, R.K., Nachtsheim, C.J.: The coordinate-exchange algorithm for constructing exact optimal experimental designs. Technometrics **37**(1), 60–69 (1995)

Michalewicz, Z., Fogel, D.B.: How to solve it: modern heuristics. Springer, New York (2004)

Mitchell, T.J.: Computer construction of "D-optimal" first-order designs. Technometrics **16**(1), 211–220 (1974)

Mitchell, T.J.: An algorithm for the construction of "D-optimal" experimental designs. Technometrics **16**(2), 203–210 (1974)

Montepiedra, G., Myers, D., Yeh, A.B.: Application of genetic algorithms to the construction of exact D-optimal designs. J. Appl. Stat. **25**(6), 817–826 (1998)

Montgomery, D.C., Jennings, C.L.: An overview of industrial screening experiments. In: Dean, A., Lewis, S. (eds.) Screening: Methods for Experimentation in Industry, Drug Discovery, and Genetics, pp. 1–20. Springer, New York (2006)

Montgomery, D.C.: Design and Analysis of Experiments. Wiley, New York (2008)

Nguyen, N.K., Miller, A.J.: A review of some exchange algorithms for constructing discrete D-optimal designs. Comput. Stat. Data Anal. **14**(4), 489–498 (1992)

Nguyen, N.K.: An algorithmic approach to constructing supersaturated designs. Technometrics **38**(1), 69–73 (1996)

Plackett, R.L., Burman, J.P.: The design of optimum multifactorial experiments. Biometrika **33**(4), 305–325 (1946)

Sörensen, K., Glover, F.: Metaheuristics. In: Gass, S., Fu, M. (eds.) Encyclopedia of Operations Research and Management Science, 3rd edn. Springer, London (2013)

Sung Jung, J., Jin Yum, B.: Construction of exact D-optimal designs by tabu search. Comput. Stat. Data Anal. **21**(2), 181–191 (1996)

Talbi, E.G.: Metaheuristics: From Design to Implementation. Wiley, New York (2009)

Trinca, L.A., Gilmour, S.G.: An algorithm for arranging response surface designs in small blocks. Comput. Stat. Data Anal. **33**(1), 25–43 (2000). Erratum **40**(3), 475 (2002)

Welch, W.J.: Algorithmic complexity: three NP-hard problems in computational statistics. J. Stat. Comput. Simul. **15**(1), 17–25 (1982)

Wu, C.F.J., Hamada, M.: Experiments: Planning, Analysis, and Parameter Design Optimization. Wiley, New York (2000)