

Enumeration and Classification of Orthogonal Arrays

Eric D. Schoen,

University of Antwerp, Belgium, and TNO Science and Industry, Delft, Netherlands

Man V.M. Nguyen,

University of Technology, Ho Chi Minh City, Vietnam

September 26, 2007

Abstract

We specify an algorithm to enumerate a minimum complete set of combinatorially non-isomorphic orthogonal arrays of given strength, run-size, and level numbers of the factors. We consider the classification of arrays according to several criteria of practical interest, differentiated according to the strength of the arrays. We exemplify classification using several series of mixed arrays, and we propose to discard arrays that are inadmissible according to the criteria.

KEY WORDS: Experimental Design; Non-Isomorphic Designs; Estimation Capacity; Minimum Forbidden Subconfiguration; Mixed Array

1 Introduction

In scientific experimentation, scientists may want to investigate the joint effect of several factors on the properties of some product or process. Frequently, there is an extensive list of candidate factors, of which only a few turn out to be active. When interactions between the active factors can be considered

negligible, it makes sense to estimate the main effects with an orthogonal array (OA) of strength 2. Formally, an OA of strength t is an $N \times n$ matrix whose i th column contains s_i different factor-levels in such a way that for any t columns, every t -tuple of levels appear equally often in the matrix (Rao 1947). Thus, in arrays with $t = 2$, for any factor A , all possible levels of any other factor appear equally often at each of the levels of A . As a consequence, the main effect of any factor can be deduced from the corresponding set of s_i means of the experimental results. This property follows only if interactions are indeed negligible. Otherwise, the means could be distorted by a particular combination of two other factors, and it is hard to disentangle the main effects from the 2-factor interactions.

OAs with $t = 3$ could be used fruitfully if there are a few substantial interactions between the experimental factors, while their identity is not known in advance. Contrary to the $t = 2$ case, the estimates of the main effects are not distorted by interactions between other factors. However, estimates of interaction components can be affected by the presence of other interaction components. Indeed, such an OA does not have every level pair of factors appearing equally often against every pair of settings of two other factors. In addition, the degrees of freedom available for the estimation of interaction components may not be sufficient to estimate them all simultaneously. So there could be a problem of interpreting an active interaction component.

Finally, OAs with $t > 3$ have orthogonal interaction components. Thus, they could be employed when many interactions are expected, at the cost of an increased run size. We refer to Hedayat, Sloane, and Stufken (1997) for a comprehensive account on properties and constructions of orthogonal arrays.

At this point, it is convenient to introduce more terminology. We use the notation $\text{OA}(N; s_1, \dots, s_n; t)$ for an OA of run size N , strength t , n factors, and level numbers s_1, \dots, s_n . These features are collectively called the parameters of an array. A mixed, or asymmetrical, array has not all its s_i equal. Pure, or symmetrical arrays have equal level numbers for all the factors. It is convenient

to use the notation $s_1^a s_2^b$ for arrays with a s_1 -level factors and b s_2 -level factors. The subsets of s_1 -level and s_2 -level columns are called the sections of the array. Finally, regular arrays have $N = s^p$ for some positive integer p . There are p s -level basic factors, and the settings of the remaining factors can be calculated by modular arithmetic from those of the basic factors.

Two arrays are said to be combinatorially isomorphic if one array can be obtained by permuting rows, columns, or factor levels of the other array. It would be highly desirable to have an enumeration method obtaining, for given parameters, a minimum complete set of OAs. Such a set has a single representative for each isomorphism class. If the factors are all qualitative, one would then have to choose the best array according to some optimality criterion, assign the factors to the columns such that prior knowledge on the factor's activities is incorporated as good as is feasible, and to assign the factor levels to the symbols in a column at random. For qualitative factors at more than two levels, one would have to consider additionally the non-equivalent ways to assign factor levels to the symbols, because the levels now are ordered.

An essential element in establishing a minimum complete set of OAs is in proving that all of the arrays are indeed non-isomorphic to each other. The problem is to avoid calculation of all possible permutations of rows, columns, and factor-levels to map one array on another array. Clark and Dean (2001) present an algorithm to test the equivalence of two-level arrays without having to calculate all these permutations. The algorithm can be extended to cases with factors at more than two levels, and indeed was used as such by Evangelaras et al. (2005).

The enumeration in a minimum complete set of OAs was addressed by Chen, Sun, and Wu (1993), Tsai, Gilmour, and Mead (2000), Xu (2002), and Sun, Li, and Ye (2002). These papers differ in the specific arrays considered and in the test of design isomorphism. Chen, Sun, and Wu (1993) is on symmetrical regular designs. The authors considered all possible extensions of a q factor array with an additional factor. For regular designs, there are a limited number of columns

to choose from. Indeed, a two-level design with 2^p runs has $2^p - (p + 1)$ such columns. The arrays with $q + 1$ factors are divided in groups according to the word-length pattern. Groups of size $n_i > 1$ were divided further in sub-groups according to the letter pattern. Here, i indexes the groups. Sub-groups with size $n_{ij} > 1$ are subjected to a complete isomorphism test, j indexing the subgroups. Only one design for each isomorphism class is retained. In this way, the authors were able to obtain a complete catalogue of all regular two-level designs of up to 64 runs. The ideas from this paper were used by later authors to obtain two-level designs with run-sizes up to 128 (Block and Mee, 2005), or three-level designs with run-sizes up to 729 (Xu, 2005).

Tsai, Gilmour, and Mead (2000) study designs for quantitative factors. They present a column-wise algorithm to obtain three-level designs of strength 2. Thus, without loss of generality, the first two columns are given. The authors then generate all columns that are orthogonal to the first column, and check whether the new columns are also orthogonal to the second one. Each of the retained columns forms an OA with the first two columns. The sets of three columns are then divided over design families according to an approximation of the mean A_s efficiency over all possible second-order models ignoring the intercept. One member of each family is kept. The authors thus avoid extensive isomorphism testing. Further factors are added in the same way as the third factor.

Sun, Li, and Ye (2002) present an algorithm to construct all non-isomorphic two-level designs of specified run-size and numbers of factors. The algorithm starts with extending all non-isomorphic designs with q factors with an additional column in all possible ways. Then, the authors calculate for each extension an extended word-length pattern. They subsequently group the $q + 1$ -factor designs. Within each group, they test for isomorphism using a newly developed algorithm based on the concept of minimal column base.

Xu (2002) presents an algorithm to generate mixed arrays of small run size. In his procedure new columns are added to an existing array by a random mech-

anism. The new array is then tested for orthogonality. There is no extensive isomorphism testing, the purpose being to construct small OAs of practical interest. Finally, the author notes that a complete search of all possible column-wise additions is computationally infeasible for N not small.

The purpose of this paper is to present a new algorithm to enumerate all non-isomorphic arrays of a given set of parameters and to suggest classification criteria bearing on suitability for practical applications. The algorithm is based on general ideas from Kaski and Östergård (2005). It handles mixed as well as pure arrays. Isomorphism testing of pairs of arrays is avoided by retaining only arrays of a specific canonical form. Further, adding complete columns and checking afterwards the orthogonality to existing columns is circumvented by element-wise addition of symbols and registers book-keeping all t -tuples of symbols.

Classification criteria should reflect the different purposes of arrays that differ in strength. We advocate some well-known criteria for strength-2 arrays, and we propose some new criteria for arrays with strength 3 or strength 4.

The rest of this paper is organized as follows. The algorithm is discussed in detail in Section 2. The discussion includes a check against cases known from the literature. In Section 3, we propose methods to classify the arrays obtained. We differentiate the classification according to the strength of the arrays. Criteria are proposed that reflect the practical potential of an array. If there are many arrays, we propose to discard those that are dominated by others according to all of the criteria. The section includes the classification for some series constructed with the algorithm. These include all combinatorially non-isomorphic $\text{OA}(18; 3^a 2^1; 2)$. Finally, there are some concluding remarks in Section 4.

2 Enumeration

The enumeration problem entails constructing a minimum complete set of OA with given run-size, strength, and level numbers of the factors. No array in such a set can be obtained from another array in the set by the joint operations of permuting columns, permuting rows, and exchanging factor-levels. The set thus does not contain combinatorially isomorphic arrays. Note that an OA can be considered as a specific kind of combinatorial object. In this section, we review general principles for algorithms that exhaustively generate all combinatorial objects of some specified class and detail an algorithm specific to orthogonal arrays. A worked example is given next. We end the section with tests against known cases and an overview of some newly generated series.

2.1 Algorithm

Our proposed algorithm implements general principles from Kaski and Östergård (2005; henceforth KO) to the specific case of generating a minimum complete set of $OA(N; s_1, \dots, s_n; t)$. We assume without loss of generality that $s_1 \geq s_2 \geq \dots \geq s_n$. We also assume that an s_i -level factor uses symbols $0, 1, \dots, (s_i - 1)$. The task at hand is called isomorph-free exhaustive generation. An individual array can be considered as a representative of its isomorphism class. The complete class can be generated from any representative by performing all possible permutations of rows, column, and factor levels. For considering an array to be used in statistical applications, these permutations are immaterial. Row permutations just change the order of the runs, column permutations change the names of the factors and level permutations, for qualitative factors, change the labels. It is obviously sufficient to keep one representative of each isomorphism class. KO suggest keeping the representative that has some canonical form. We propose to keep arrays only if they are lexicographically minimum in columns, according to Definition 1.

Definition 1. Consider two $OA(N; s_1, \dots, s_n; t)$, say, A_1 and A_2 . Write a_1 and a_2 for the $N.n$ -tuples obtained by concatenating the columns of A_1 and of A_2 , respectively. Let the elements of these tuples be denoted with $a_{ij}, i = 1, 2; j = 1 \dots N.n$. A_1 is lexicographically less than A_2 , denoted $A_1 < A_2$, if there is a $k \leq N.n$ such that $a_{1j} = a_{2j}$ for $j = 1 \dots k - 1$ and $a_{1k} < a_{2k}$. A_1 is lexicographically minimum in columns if no other array from its isomorphism class is smaller.

Hereafter, we will omit the additional reference to the columns from the specifications of lexicographical ordering.

The algorithm searches for lexicographically minimum arrays, the domain of the search being all $OA(N; s_1, \dots, s_q; t)$ with $t < q \leq n$, joined with the unique $OA(N; s_1, \dots, s_t; t)$ that is lexicographically minimum. The latter array will be called the root node R . We also define a search tree to be a rooted tree whose nodes are objects in the domain of the search. Two nodes are joined by an edge if and only if they are related by one search step. This step is the extension of a particular $OA(N; s_1, \dots, s_q; t)$ to an $OA(N; s_1, \dots, s_{q+1}; t)$. So the parent and the child have q identical columns, and the child has an additional column with s_{q+1} factor levels. We will denote with $C(X)$ the set of all child nodes of a node X and with $p(X)$ the parent node of X .

The algorithm uses a depth-first traversal of the search tree. Consider a node X . A pseudo-code description of a depth-first approach is:

1. DEPTH FIRST (X)
2. for all $Y \in C(X)$ do
3. DEPTH FIRST (Y)
4. end for

As yet, the way $C(X)$ is constructed from X is left open. We now consider the part of the algorithm that extends an array X with $q \geq t$ columns to all

$Y \in C(X)$ with $c = q + 1$ columns. We denote the elements of the additional column with v_{rc} . A completed column, v_c , is called a solution. A pseudo-code description follows.

Extension algorithm. Input: X ; output: $C(X)$.

1. Let $c := q + 1$, and $r := 1$.
2. Without loss of generality, let $v_{1c} = 0$.
3. Procedure ELEMENT ADDITION(v_{1c}, \dots, v_{rc})
4. if v_{1c}, \dots, v_{rc} is a solution, then
5. Append the solution to X .
6. Report $X|v_c$.
7. end if
8. if $r = N$, then stop
9. If $v_{rj} = v_{(r+1)j}$ for all $j \in \{1, \dots, (c - 1)\}$, then choose $v_{(r+1)c}$ from $\{v_{rc}, \dots, (s_c - 1)\}$. Otherwise, choose $v_{(r+1)c}$ from $\{0, \dots, (s_c - 1)\}$.
10. Check for each choice of $v_{(r+1)c}$ its compatibility with the strength t of the array.
11. Collect compatible choices in the set $A_{r+1}(v_{1c}, \dots, v_{rc})$.
12. for all $a_{r+1} \in A_{r+1}(v_{1c}, \dots, v_{rc})$ do
13. ELEMENT ADDITION($v_{1c}, \dots, v_{(r+1)c}$)
14. end for
15. end procedure
16. end algorithm

The following theorem ensures that $\rho(X) \in C(p(X))$, where $\rho(X)$ is the canonical form of X , and $p(X)$ is in canonical form.

Theorem 1 *The extension algorithm produces nodes X for which $\rho(X) \in C(p(X))$, if $p(X)$ is in canonical form.*

Proof. Consider the parent node $p(X)$. A lexicographically minimum child node must have $v_{1c} = 0$. This is ensured by step 2. Step 9 prevents generation of an array that is lexicographically higher than is necessary. Otherwise, it permits all possible choices of elements that are compatible with the required strength. \square

Suppose for the moment that we have a way of testing whether a node is lexicographically minimum. The algorithm considers $C(X)$ only if X is indeed of this form. So an array is augmented only if its parent has the canonical form. KO call this way of generating the arrays orderly generation. Their Theorem 4.20 states that if the orderly generation is implemented on a search tree that satisfy two conditions, it reports exactly one node from each isomorphism class of nodes. The conditions are (1) the search tree has nodes X for which $\rho(X) \in C(p(X))$, and (2) for every nonroot node X in the search tree that is in canonical form, $p(X)$ is also in canonical form. Condition (1) is ensured by Theorem 1 above. For condition (2), we note that the objects in the search domain can be considered as $N.n$ tuples over an alphabet of $\max\{s_i\}$ symbols and the additional symbol \diamond for entries in columns yet undefined. The lexicographical order of symbols from the smallest to the largest element is taken to be $0, 1, \dots, \max\{s_i\}, \diamond$. The lexicographic significance of a position in the tuple is its order number. We rephrase Theorem 4.24 of KO as

Theorem 2 *For every nonroot node x and for a search tree algorithm constructing $N.n$ -tuples in order of lexicographic significance: if $p(x) \neq \rho(p(x))$ then $x \neq \rho(x)$.*

By noting that our algorithm does construct the tuples in increasing order of lexicographical significance, we see that Theorem 2 applies. Condition (2) above is just the reversal of the implication statement in the theorem. We conclude that the algorithm reports exactly one node from each isomorphism class of nodes, and therefore, exactly one array of each isomorphism class.

It remains to be discussed how to test whether a node is lexicographically minimum. Suppose that the algorithm produces a node A , a specific $\text{OA}(N; s_1, \dots, s_q; t)$, with $q \leq n$. We want to extend this array only if it is lexicographically minimum. We propose the following

Lexicographic test

1. For each row r of A , enumerate one by one all level permutations h such that $c_{rj}^{h(A)} = 0$ for all $1 \leq j \leq k$.
2. For each $h(A)$, enumerate all permutations g such that column i is permuted with other columns with s_i levels, for $1 \leq i \leq k$.
3. For each permuted array $g.h(A)$, apply the unique row permutation $s(g.h(A))$ that results in the lexicographically smallest image of $g.h(A)$.
4. If $s(g.h(A)) < A$ for some h and g , then discard A .
5. If $A < s(g.h(A))$ for all permutations g and h , then keep A .

An important issue is the number of permutations to consider for the lexicographic test. Using the notation $\text{OA}(N; s_1^{q_1} s_2^{q_2} \dots s_v^{q_v}; t)$ with $s_1 > s_2 > \dots > s_v$, we observe that each array that has been kept has been subjected to the following numbers of permutations:

$$|g| = \prod_{i=1}^v q_i!$$

$$|h| = N \cdot \prod_{i=1}^v ((s_i - 1)!)^{q_i}$$

$$|g.h| = N \cdot \prod_{i=1}^v q_i! ((s_i - 1)!)^{q_i}$$

It is obvious that for larger problems the computation time becomes huge by just considering the lexicographic test. On the other hand, the test reduces the search space by pruning arrays that are not lexicographically minimum at the end of each column. Also, because of the test we obtain solutions that are unique in their isomorphic class.

The main algorithm can now be formulated as follows.

Construction of a minimum complete set of OA($N; s_1, \dots, s_n; t$)

1. Construct the root node R . Set $X = R$.
2. Set $q = nc(X)$, nc denoting the number of columns.
3. If $q = n$ then stop. Else
4. Invoke the extension algorithm; input: X ; output: $C(X)$.
5. Set $d = |C(X)|$.
6. If $d = 0$, then stop. Else
7. For $i = 1 \dots d$ do
8. Invoke lexicographic test; input: $C_i(X)$.
9. If $C_i(X) = \rho(C_i(X))$ then set $X = C_i(X)$ and return to 3.
10. End for.

2.2 A worked example

We now illustrate the various stages of the algorithm with the construction of a minimum complete set of OA(16; $4^1 2^3$; 3); see Table 1. The algorithm starts with construction of the root node R shown in the left panel of the table. From this node, the extension algorithm starts. The first element of the third two-level factor is taken to be 0. For the second element, we have the choice between a 0 and a 1. The second panel of the Table 1 shows the first option. It is a faulty one, because there should be exactly one 0/0/0 triple of the four-level

factor, the first two-level factor, and the third two-level factor. So the set $A_2(0)$ has just one element namely $(0, 1)$.

The first and only set $A_{r+1}(v_{1c}, \dots, v_{rc})$ to have more than one element is $A_5(0, 1, 1, 0)$, with elements $(0, 1, 1, 0, 0)$, and $(0, 1, 1, 0, 1)$, respectively. From that point onward, there are only unique compatible choices. The resulting arrays are given as Array 1 and Array 2 in Table 1.

Upon completion of an array, the lexicographic test is applied. Array 1 is lexicographically minimum. The total number of permutations needed to establish this fact is $16 \cdot 3! \cdot 3! = 576$. Array 2 can be turned into array 1 by permuting the levels 1 and 2 of the four-level factor and sorting of the rows. So this array is discarded. We conclude that there is a unique $\text{OA}(16; 4^1 2^3; 3)$.

Suppose now that we want to construct a minimum complete set of $\text{OA}(16; 4^1 2^4; 3)$. The root of the search tree is the same as the above one. There are two child nodes $C(X)$, viz. Array 1 and Array 2. Array 2 is not subjected to the extension algorithm because it is not lexicographically minimum. Array 1 is subjected to extension. The first two elements of the new column must be

Table 1: Construction of $\text{OA}(16; 4^1 2^3; 3)$

Root	false step	Array 1	Array 2
0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
0 0 1	0 0 1 0	0 0 1 1	0 0 1 1
0 1 0	0 1 0	0 1 0 1	0 1 0 1
0 1 1	0 1 1	0 1 1 0	0 1 1 0
1 0 0	1 0 0	1 0 0 0	1 0 0 1
1 0 1	1 0 1	1 0 1 1	1 0 1 0
1 1 0	1 1 0	1 1 0 1	1 1 0 0
1 1 1	1 1 1	1 1 1 0	1 1 1 1
2 0 0	2 0 0	2 0 0 1	2 0 0 0
2 0 1	2 0 1	2 0 1 0	2 0 1 1
2 1 0	2 1 0	2 1 0 0	2 1 0 1
2 1 1	2 1 1	2 1 1 1	2 1 1 0
3 0 0	3 0 0	3 0 0 1	3 0 0 1
3 0 1	3 0 1	3 0 1 0	3 0 1 0
3 1 0	3 1 0	3 1 0 0	3 1 0 0
3 1 1	3 1 1	3 1 1 1	3 1 1 1

0 and 1, respectively. The third element cannot be a 0, because there should be exactly one 0/0/0 triple of the four-level factor, the first two-level factor, and the fourth two-level factor. Neither can it be a 1, because there should be exactly one 0/1/1 triple of the four-level factor, the third two-level factor, and the fourth two-level factor. Here, the extension algorithm stops. We conclude that $OA(16; 4^1 2^4; 3)$ does not exist. In fact, this exemplifies an easy-to-prove theorem:

Theorem 3 *If there is a unique $OA(N; s_1, \dots, s_n; t)$, it is not possible to extend the array with an additional column that has level numbers equal to one of the original columns, while maintaining strength t .*

2.3 Discussion

The proposed algorithm features element-wise addition of symbols in a column, isomorphism testing performed on single arrays and a depth-first approach. These features jointly define for what cases the algorithm is expected to be useful. First, the element wise addition has advantages over an addition of a new column balanced for the first $t - 1$ columns (Tsai, Gilmour, and Mead, 2000; Sun, Li, and Ye, 2002) for growing run size. For example, the addition of two-level columns to $OA(24; 3^1 2^2; 3)$ requires evaluation of 46656 columns. For the $OA(48; 3^1 2^2; 3)$, this number grows to more than 117 billion. For adding a three-level column to $OA(54; 3^3; 3)$ the number grows to 3.87×10^{17} . Elementwise addition is clearly useful in these cases.

The isomorphism testing performed on single arrays has the advantage that it permits parallelization of the algorithm. Given a set of, say, m arrays that have to be subjected to isomorphism testing, a pairwise testing procedure could involve $m - 1 \leq c \leq 0.5m(m - 1)$ tests. For the single-array testing, $c = m$. On the other hand, the single-array testing requires that the testing is comprehensive, while the pairwise testing need only to be applied within groups that are homogeneous with respect to initial classification criteria. The need for

comprehensive testing renders single-array testing impractical for arrays with many factors. For example, each lexicographically minimum $OA(27; 3^{12}; 2)$ has to be subjected to 5.30×10^{13} permutations to determine whether it is indeed of canonical form.

From the above paragraphs, we conclude that the proposed algorithm is likely to be useful for moderate to large run sizes in conjunction with a moderate number of factors. This formulation is deliberately vague, because much also depends on computing resources, and the strategy applied when an array with an added column has passed the test. Our present algorithm then proceeds with adding additional columns as it is depth-first. One could change priorities and make a breadth-first algorithm as used, for example by Chen, Sun, and Wu (1993) for regular two-level designs.

2.4 Known and new cases

We implemented the algorithm in a C program. One peculiar feature of C is in its integer representation taking 32 bits. One bit is used to store the sign; there thus remain 31 bits for the value itself. This implies that we can use integers up to at most 2^{31} . When it comes to permuting columns in isomorphism testing, we can handle at most 12 columns with the same factor-levels, as $12! < 2^{31} < 13!$.

We tested our algorithm against three known cases. First, Li (2006) found 55 distinct $OA(16; 2^7; 2)$. Our algorithm reproduced this number of cases. All cases are given explicitly in Sun, Li, and Ye (2002). From the cases given by these authors, it can be established that there is a single $OA(16; 2^7; 3)$. Likewise, our algorithm returned a single array of this type.

As a second test, we were able to reproduce the finding of Hedayat, Seiden, and Stufken (1997) that there are exactly four isomorphism classes of $OA(54; 3^5; 3)$.

Finally, we confirm Xu, Cheng, and Wu (2004) in their finding by random search that there are three distinct $OA(18; 3^7; 2)$.

We generated some new series of $OA(N; s_1^a s_2^b; t)$ for $t = 2, 3, 4$, to be able to

discuss our classification criteria. Without risk of confusion, a specific series will be designated $N.a.b$, and the i th member of a series with $N.a.b.i$. An overview is given in Table 2. We generated all combinatorially non-isomorphic OA(18; $3^a 2^1$; 2). Until now, the L_{18} (see, e.g., Wu and Hamada, 2000) and projections thereof into less factors were the only known members of these series. We observe a considerable size of the minimum complete subsets for 3 up to 6 three-level factors. Some aspects of the classification are discussed in Section 3. A comprehensive classification of all 18-run orthogonal arrays, including projection properties into arrays with a smaller number of factors, is given in a separate paper (Schoen, 2007).

The strength-3 arrays in Table 2 are all $3^a 2^b$ arrays. None of these arrays can be extended with additional three-level or two-level factors while at the same time retaining strength 3. This fact, and the numbers of isomorphism classes of the 48-run and 54-run cases were given earlier in Brouwer, Cohen, and Nguyen (2006). The only other literature reference on mixed two-and-three-level designs known to us is Connor and Young (1961). The designs of these authors permit estimation of all main effects and two-factor interactions. Our series have smaller run-sizes than the corresponding designs of Connor and Young. Their 3^{12^4} and 3^{12^9} arrays have run sizes of 36 and 128 and strengths of 1 and 0, respectively.

Table 2: Some new series of orthogonal arrays

t	N	type	# ni arrays
2	18	$3^7 2^1$	3
		$3^6 2^1$	12
		$3^5 2^1$	19
		$3^4 2^1$	48
		$3^3 2^1$	15
		$3^2 2^1$	3
3	24	3^{12^4}	3
		3^{12^9}	3
		$3^5 2^1$	4
4	128	$4^3 2^3$	6
		$4^2 2^6$	77
		$4^1 2^9$	275

So the first of these arrays has one third of the observations allotted to each level of the three level factor, and one half of the observations allotted to each level of any two-level factor. There is no further balance. The second of the arrays has no level-balance at all. The $3^5 2^1$ array of Connor and Young has strength 4 and run size 162; this is just a duplicated regular $OA(81; 3^5; 4)$. So our arrays could be a suitable alternative to the Connor and Young designs; statistical properties are discussed later in this paper.

Similar to the tabulated strength-3 series, the tabulated strength-4 arrays cannot be extended with an additional four-level or two-level factor while retaining strength 4. We chose the 128-run series, because series for all possible smaller run-sizes had at least one solution that can easily be constructed from regular designs.

3 Classification

Our primary purpose to construct minimum complete series of orthogonal arrays is to find arrays useful as an experimental design to study the effect of controllable factors. The factors involved may be quantitative or qualitative. Our construction algorithm is focused on qualitative factors because level permutations of a factor are considered to result in equivalent designs. So it makes sense that classification criteria for our arrays should reflect the quality of an array when used as a practical design for qualitative factors. Literature criteria for two-level designs include the well-known aberration criterion of Fries and Hunter (1980) for regular designs, and the generalized resolution of Deng and Tang (2002) for irregular designs. Criteria for quantitative factors include the Q criterion of Tsai, Mead and Gilmour (2000), and the criteria based on the β -WLP of Cheng and Ye (2004). Criteria for general factorial designs were proposed by Xu and Wu (2001), and criteria focused on model discrimination were proposed by Srivastava (1975) and Jones et al. (2007); see also the references contained in the latter article.

In this section, we propose to differentiate the criteria according to the strength of the array. There will be a few simple criteria for each strength, and we should want to restrict attention to the arrays that have good combined properties regarding these criteria. Following Sun et al. (1997), we discard inadmissible arrays. An array is inadmissible according to c criteria if there is another array that is strictly better according to at least one of the criteria and equally good according to the remaining criteria. Otherwise, the array is called admissible.

In the remainder of this section, we introduce the criteria for strength 2, 3 and 4 separately. We study the series of arrays from Table 2 with the proposed criteria.

3.1 Criteria for $t = 2$

Arrays of strength 2 could fruitfully be used to detect active main effects of the factors. Here, it is important that two-factor interactions (2fis) should not hamper the detection procedure. We adopt two literature criteria to quantify this notion. The first one is the number of words of length 3 in the generalized word-length pattern (GWLP) defined by Xu and Wu (2001). The GLWP of an array with n factors is a vector (A_1, A_2, \dots, A_n) . The A_i can be viewed as the sum of all squared and standardized inner products of two columns measuring

Table 3: Admissible OA(18; $3^a 2^1$; 2)

Array	A_3	Projected A_3 frequency					
		0	$4/9$	$1/2$	$2/3$	1	2
18.7.1.1	28	9	9	16	21	0	1
18.6.1.1	16	6	0	20	9	0	0
18.5.1.1	$8^{1/2}$	7	0	7	3	3	0
18.5.1.3	$8^{2/3}$	4	6	4	6	0	0
18.4.1.1	$3^{1/2}$	6	0	1	0	3	0
18.4.1.4	$3^{7/9}$	2	4	4	0	0	0
18.3.1.1	$1/2$	3	0	1	0	0	0

NOTE: projected A_3 frequency gives the number of 3-column subarrays that have the indicated A_3 .

a j -factor and a $(i - j)$ -factor interaction, respectively. For all designs, the A_i are independent of the choice of orthonormal contrasts used to calculate the interactions (Dey and Mukerjee, 1999).

For arrays of strength 2, $A_1 = A_2 = 0$. Contamination of main effects with 2fis is measured by A_3 , and, in general, we prefer arrays with a minimum value of this quantity. Xu, Cheng, and Wu (2004) note that there may be several non-isomorphic arrays with identical GWLP, and thus with identical A_3 -values. For example, we obtained three non-isomorphic $3^7 2^1$ arrays. All three arrays have the same GWLP, with $A_3 = 28$. In order to discriminate designs with identical GWLP, the aforementioned authors propose the projection aberration criterion. This criterion uses the A_3 values of the 3-factor projections. For a $3^a 2^1$ array, we have $(a + 1)a(a - 1)/6$ 3-factor projections. The possible A_3 values of the projections appear to be 0, $4/9$, $1/2$, $2/3$, 1 or 2. The A_3 values are ordered according to a decreasing desirability. A value of 2 implies a complete aliasing of a main effect with either the $Y + Z \pmod{3}$ or the $Y + 2Z \pmod{3}$ component of the 2fi among the other two factors of the three-factor projection. A value of 0 implies orthogonality. The projection aberration criterion sorts designs according to their projection frequency starting from the worst projection A_3 -value. For example, Table 3 shows that designs 18.5.1.1, and 18.5.3 have projection frequencies of $(7\ 0\ 7\ 3\ 3\ 0)$, and $(4\ 6\ 4\ 6\ 0\ 0)$, respectively. The first array is slightly better in terms of total A_3 . However, three of its 3-factor subarrays have $A_3 = 1$. This is worse than is the case with 18.5.1.3 that has none of its subarrays with $A_3 = 1$.

Table 3 presents the classification for all admissible $OA(18; 3^a 2^1; 2)$ with $3 \leq a \leq 7$. For $a = 5$ and $a = 4$, there are two admissible arrays; the remaining cases each have a single admissible array. The arrays themselves are given in the Appendix. No array is given for $a = 2$, because the only admissible array is just the full factorial. It is interesting to note that the $3^7 2^1$ presented there is not isomorphic to the widely used L_{18} . Further details are given by Schoen (2007).

Jones et al. (2007) give criteria to quantify a design's ability to distinguish among models from some specified family. In our case, such a family would consist of all the main effects and a few 2fis. However, an interaction among qualitative three-level factors takes 4 degrees of freedom. So we believe that the primary concern here should be with the contamination between main effects and 2fis as measured with our proposed criteria. We refer to the aforementioned paper for the case of qualitative three-level factors and projections of the L_{18} .

3.2 Criteria for $t = 3$

Consider an $OA(N; s_1 s_2 \cdots s_n; 3)$, \mathcal{F} , say. Call the columns in \mathcal{F} 'original columns'. Replace any original column of s_i levels with $s_i - 1$ orthogonal columns, and call these the 'main effect columns'. Add to the left a column $\mathbf{1}$ and call the matrix of the new set of columns F_1 . Construct the p -extended model matrix F_p by extending F_1 with columns formed by the entry-wise products of $2, \dots, p$ of the main effect columns for $p = 2, \dots, n$. Do not form products of main effect columns obtained from the same original column.

As the arrays are orthogonal, we know that F_1 has full rank. In orthogonal arrays of strength 3, the columns in F_1 are orthogonal to each other and to the columns in the 2-extended model after excluding the columns of the 1-extended model, $F_2 \setminus F_1$. So contamination of main effects with 2fis is of no concern here. If we can ignore higher-order interactions, main effects can be identified with standard F tests or with methods judging the main effects with a robust estimate of the standard error constructed from the effects themselves (Lenth 1989, Schoen and Kaul 2000, Loeppky and Sitter 2001).

The challenge in strength-3 arrays is to identify the active 2fis collected in $F_2 \setminus F_1$. So classification criteria for strength-3 arrays in a minimum complete set should focus on this issue. We propose three simple criteria and illustrate with the strength-3 arrays in Table 2, augmented with the $OA(54; 3^5; 3)$ taken from Hedayat, Seiden and Stufken (1997). We included this series as the statistical properties of the arrays have not been studied previously. Classification of the

strength-3 arrays is given in Table 4.

First, we propose to include A_4 as a classification criterion of strength-3 arrays; see the explanation in the previous subsection. We note that minimization of this criterion minimizes the sum of squared correlations among 2fi components. As shown in Table 4, the criterion does not uniquely identify the arrays within a series. Thus, for $t = 3$, we would also want to consider other criteria.

Second, we propose to include the rank $n_2 = r(F_2) - r(F_1)$ of the $F_2 \setminus F_1$ matrix. This quantity specifies the maximum number of estimable components of 2fis in a model based on the array. It may be refined further by looking at the ranks for meaningful subsets of the 2fis. For example, one could be specifically interested in interactions among 2-level factors.

In general, we would prefer arrays with large n_2 . Table 4 shows the values for the series of strength-3 arrays considered in this paper. It is remarkable that 54.5.0.4 has almost all its interaction components estimable. Further, both 24.1.4.1 and 24.1.4.3 have all components of 2fi estimable. In view of the strength of the arrays, the interaction estimates must be correlated. As noted

Table 4: Summary of strength-3 arrays

Array	n_2	A_4	MFS
24.1.4.1	14(0)	$7/9$	0
24.1.4.2	11(3)	1	3
24.1.4.3	14(0)	$1/9$	0
48.1.9.1	36(18)	$8^2/9$	0 0 0 36 4
48.1.9.2		$8^2/9$	0 0 0 48 16
48.1.9.3		$8^5/9$	0 0 0 5 4
54.5.0.1	31(9)	$3^{1/18}$	0 0 0 0 25
54.5.0.2	36(4)		
54.5.0.3	35(5)	3	1 0 0 0 4
54.5.0.4	39(1)		0 0 1 0 0
54.5.1.1	41(9)	$5^2/3$	1 0 1 0 4 0 18
54.5.1.2			1 0 2 0 4 0 16
54.5.1.3			0 0 2 2 0 0 0 0 3
54.5.1.4			0 0 4 0 0 0 0 0 3

NOTES: n_2 gives degrees of freedom for 2fi components. Figures between brackets: non-estimable df of full 2fi models. MFS given for models with 2-10 2fi for 3^{12^4} series and 2-6 2fi for remaining series.

by Hedayat, Seiden, and Stufken (1997), the first two 3^5 arrays have two and one duplicate runs, respectively. So there are 2 and 1 degree of freedom, respectively, to estimate pure error. As this is an exception rather than the rule, we do not propose to include a classification criterion based on pure error degrees of freedom. Finally, we note that the joint criteria of n_2 and A_4 fail to discriminate the $3^5 2^1$ arrays and two of the $3^1 2^9$ arrays.

To define the third classification criterion, consider a model \mathcal{M} containing all components of k 2fis (2fis). The set of all these components is called a minimum forbidden subconfiguration (MFS) if

1. \mathcal{M} contains at least 1 non-estimable component.
2. Deleting all components of any of the k 2fis would result in an estimable model.

Augmentation of \mathcal{M} with other interactions also results in a model that is not fully estimable. For an orthogonal array of strength 3, all models with a single 2fi are estimable. The set of all MFS for $k = 2, \dots, \binom{n}{2}$ permit an assessment of models that can be estimated with the particular array. The set can be found by calculating the ranks of submatrices of $F_2 \setminus F_1$. Note that this can involve heavy computations for large n . Note also, that MFS is a vector-valued criterion. We want to minimize MFS sequentially, from the leftmost element onward.

Tabel 4 gives MFS values for the designs of special interest. For the 24.1.4 cases, we have 10 possible interactions, and we considered values of k to 10. The figures bear on MFS for $k = 2$. The MFS for all remaining values of k was 0. Of the three arrays, only array 24.1.4.3 is admissible if we use n_2 , A_4 and MFS as criteria.

For the 48.1.9 cases, there are 45 possible interactions. We considered values of k from 2 up to 6. Array 48.1.9.2 is inadmissible under the joint criteria of A_4 , n_2 , and MFS. The other two arrays are admissible under these criteria.

For the 54.5.0 and 54.5.1 cases, we have 10 and 15 possible interactions, respectively. We considered values of k up to 10. There are two admissible

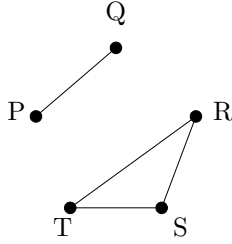


Figure 1: Minimum forbidden subconfiguration for array 54.5.0.4

3^5 arrays. The first one, 54.5.0.3, has 36 out of the 40 2fi degrees of freedom estimable. This is opposed to the 39 components in the fourth array. However, the MFS vector of the fourth array indicates a non-estimable 4-2fi model, where all 5-2fi models of the second array are estimable.

The first two $3^5 2^1$ are derived from 54.5.0.3, and the remaining two from 54.5.0.4. The single admissible $3^5 2^1$ array is the third one of the series.

The MFS criterion is closely related to the concept of estimation capacity (Cheng et al., 1999). The estimation capacity for v 2fis (EC_v) is defined as the fraction of all $\binom{c}{v}$ v -interaction models that are estimable. Here, $c = 0.5n(n-1)$ is the total number of interactions, not of interaction components. We favor MFS, because it is easier to see for what numbers v new linear restrictions apply.

The MFS criterion also gives information on model robustness of an array. Model robustness is a design's ability to identify the true model from a family of competing models. Conditions for which one can do so for all the models containing main effects and k interactions were identified by Srivastava (1975). Assuming observations without error, he showed that a necessary and sufficient condition is the estimability of all models containing $2k$ interactions. (His actual formulation was more general. Here we apply his results to strength-3 arrays assuming that we always want to estimate all the main effects.) A design that fulfills the condition is said to be strongly resolvable with resolving power k .

For our designs, we can identify k as the minimum v for which $MFS_{2v} > 0$.

So, for example, we can see that all $OA(48; 3^{12^9}; 3)$ have $k = 2$ and it is possible to distinguish among all models with 2 2fi. However, models with more than k interactions may still be distinguishable. By way of an illustration, Figure 1 shows the single MFS for array 54.5.0.4 (the array can be obtained by deleting the two-level column of array 54.5.1.3 in the Appendix). The MFS contains the interactions PQ, QT, TP and RS. If we can assume absence of one of these interactions, all nine remaining interactions can be estimated.

For the purpose of classification of designs, MFS is more refined than k because it is a vector-valued measure. It identifies families of models that have an EC of 100% and it permits a more detailed assessment of arrays for cases where EC is smaller. One may follow up with studying model robustness in families with EC = 100% proposed by Jones et al. (2007).

In our definitions of MFS and EC, we consider estimability of models \mathcal{M} containing all components of v 2fis. If an array has at least one factor at more than two levels, we could also consider models for which some of the individual interaction components are estimable. We prefer the former option, because the designs considered here will be mainly used for categorical factors. It is unlikely that an interaction between such factors can be modeled using a subset of the interaction components.

Our preference for models containing full sets of interaction components only applies to irregular designs - as are the designs of special interest considered here. For regular designs, components of interaction are defined by modular arithmetic (see, e.g., Wu and Hamada 2000). While it remains unlikely that an interaction between categorical factors can be fully modeled using only one component, the components form mutually orthogonal sets whose activity can be judged by standard methods.

3.3 Criteria for $t = 4$

For arrays of strength 4, all main effects and all 2fi are estimable orthogonally from each other. Taking the A_5 value of the arrays as one of the classification

criteria turns out to be uninformative for the cases we have studied. Instead, we propose to classify these arrays according to MFS spectra and n_2 spectra of the sub-arrays that result from taking the rows that have a particular level of one of the factors. We exemplify with the classification of the six $OA(128; 4^3 2^3; 4)$ in Table 5. There are $3 \times 4 = 12$ possible sub-arrays resulting from taking one particular level of a four-level factor. These sub-arrays have either an MFS of $(5 \ 2 \ 0)$ or of $(9 \ 0 \ 0)$, where the first MFS is the better one. The Table shows MFS spectra, indicating how many of the 12 subarrays have the first MFS and how many have the second one. As each of the subarrays have 32 runs, we indicate these spectra with $MFS(32)$. The arrays 2 and 4 both have the best spectrum. Apparently the information needed to estimate all the 2fi is spread out evenly over the subarrays.

There are $3 \times 2 = 6$ possible subarrays that have a level of some two-level factor in common. Here, there are seven possible MFS vectors, as indicated in the table. The best spectrum is the one with the right-most non-zero entry as much as possible to the left. This implies minimization of the less favorable MFS vectors. Array 4 has the best spectrum.

We also calculated n_2 spectra for the arrays. Als sub-arrays with 32 runs have 12 df for estimation of 2fi. For 64-run subarrays, possible values are 40, 42, 43, 44 and 45, respectively. Here, we want to have the left-most non-zero

Table 5: Summary of all non-isomorphic $OA(128; 4^3 2^3; 4)$

Array	MFS(32)	MFS(64)	$n_2(64)$
1	0 12	0 0 0 0 0 0 6	6 0 0 0 0
2	12 0	0 2 0 0 4 0 0	2 0 4 0 0
3	4 8	0 0 0 2 0 4 0	0 4 0 2 0
4	12 0	2 0 4 0 0 0 0	0 0 0 2 4
5	6 6	0 2 0 0 4 0 0	0 0 6 0 0
6	8 4	2 0 0 0 4 0 0	0 0 4 2 0

NOTES: array 4 is admissible; all arrays have WLP (0,0,3); MFS(32) spectra for 2-4 2fi are $(5 \ 2 \ 0)$ and $(9 \ 0 \ 0)$; all subarrays with $N = 32$ have $n_2 = 12$; MFS(64) spectra for 2-4 2fi are $(0 \ 1 \ 0)$, $(0 \ 3 \ 0)$, $(1 \ 0 \ 0)$, $(2 \ 0 \ 0)$, $(3 \ 0 \ 0)$, $(4 \ 0 \ 0)$, and $(6 \ 0 \ 0)$; $n_2(64)$ values are 40, 42, 43, 44, and 45.

entry as much as is possible to the right. Again, array 4 is best. In fact, this is the only admissible array in the series under the joint criteria of MFS(32), MFS(64), $n_2(32)$, and $n_2(64)$.

We now briefly discuss classification of the two remaining series of OA(128; $4^a 2^b$; 4) with $a, b \geq 1$. Table 6 summarizes results for admissible arrays with $a = 2$ and $b = 6$. There are 11 such arrays. None of the arrays are dominated by any of the other arrays. For example, array 128.2.6.14 has all its 8 32-run subarrays with an MFS of (0 14 8), while 128.2.6.15 has four sub-arrays with this MFS and four with (3 20 0). Here, the first array is better than the second one. This is also the case if we judge from the MFS(64). However, the $n_2(64)$ -spectrum of the first array has its first non-zero entry on the first position. For the second array, the first non-zero entry is on the third position. Here, the second array is better than the first one.

It is hard to tell which of the admissible arrays in Table 6 should be chosen as an experimental design. For the $n_2(32)$ spectra the possible n_2 values are 20 and 21 df, respectively. These values are too close to help in choosing an array. The other spectra are really different. The best MFS(32) is the one of array

Table 6: Summary of all admissible OA(128; $4^2 2^6$; 4)

Array	MFS(32)	$n_2(32)$	MFS(64)	$n_2(64)$
14	8 0 0 0	8 0	0 0 8 0 4 0 0 0 0 0 0 0	4 0 0 0 8 0 0 0
15	4 4 0 0	8 0	4 0 0 0 0 0 0 8 0 0 0 0	0 0 8 0 0 0 4 0
30	0 0 0 8	0 8	0 0 0 0 0 2 4 0 0 0 2 4	0 8 2 0 2 0 0 0
39	4 2 2 0	8 0	12 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 8 4 0
47	0 4 2 2	6 2	2 0 0 2 0 0 0 8 0 0 0 0	0 0 0 0 8 0 2 2
56	0 0 4 4	4 4	0 0 0 4 0 0 0 8 0 0 0 0	0 0 0 0 8 0 4 0
68	4 0 0 4	4 4	0 0 0 4 0 0 0 0 8 0 0 0	0 0 4 4 0 0 4 0
69	2 2 0 4	4 4	0 0 0 4 0 0 0 8 0 0 0 0	0 0 0 0 12 0 0 0
73	2 2 2 2	6 2	0 2 0 2 0 0 0 8 0 0 0 0	0 0 0 0 12 0 0 0
74	4 2 0 2	6 2	0 2 0 2 0 0 0 4 4 0 0 0	0 0 4 0 6 2 0 0
77	2 2 4 0	8 0	12 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 4 8

NOTES: admissible and inadmissible arrays all have WLP (0,0,6,1); MFS(32) spectra for 2-4 2fi are (0 14 8), (3 20 0), (6 6 8), and (9 4 0); $n_2(32)$ values are 20 and 21; MFS(64) spectra for 2-4 2fi are (0 0 0), (0 0 1), (0 0 2), (0 1 0), (0 2 8), (0 4 1), (0 7 0), (3 0 0), (3 0 1), (3 1 0), (3 2 0), and (3 2 2); $n_2(64)$ values are 41, 42, 44, 45, 46, 47, 48, and 49.

128.2.6.14; the best MFS(64) and $n_2(64)$ spectra are those of array 128.2.6.77. If there is a clear interest either in 32-run subarrays or in 64-run subarrays, these are the arrays to choose from. This could be the case if the subarrays are used as blocks of the total array.

It remains to discuss the 4^{129} series. Of the 32-run subarrays, there are 7 different MFS vectors and 3 different n_2 values. Of the 64-run subarrays, there are 48 different MFS vectors and 12 different n_2 values. There is just a single admissible array out of the 275 arrays. All its 4 32-run subarrays have MFS (0 0 18) and $n_2 = 22$. All its 18 64-run subarrays have MFS (0 0 0). Fourteen of these subarrays have $n_2 = 54$; the remaining ones have $n_2 = 51$.

4 Conclusion

This paper features enumeration of series of $\text{OA}(N; s_1 s_2 \cdots s_n; t)$ and selection of an array from such a series to be used as an experimental design. We formulated a simple algorithm for the enumeration of a complete series. We believe the algorithm to be particularly useful for moderate to large run sizes in conjunction with a small to moderate number of factors. We further proposed classification criteria that are easy to calculate. We differentiate the criteria according to the strength of the array. The criteria reflect the array's potential to be used for practical experimentation. They permit reduction of large sets of non-isomorphic arrays to much smaller sets of admissible arrays. However, if other criteria are to be considered in conjunction with those proposed here, the list of admissible designs will grow.

Using an algorithm to generate series of arrays gives little information on succinct ways to construct individual arrays. Knowledge of an explicit construction method may well lead to more insight into the best way to analyse the results of an experiment. We would therefore welcome research on such methods. All arrays discussed in this paper are obtainable electronically from the first author.

Acknowledgements

The first draft of this paper was written when both authors were at Eindhoven University of Technology in the Netherlands. The authors are grateful to Andries Brouwer for a computer implementation of the algorithm handling specific cases, to Arjeh Cohen and the late Jan Dijkstra for their encouragement, and to Ruben Snepvangers for a computer implementation of the algorithm handling general cases.

Appendix: Selected Orthogonal Arrays

Table 7: OA(18; $3^a 2^1; 2$) admissible according to A_3 and projected A_3 frequency

18.7.1.1	18.6.1.1	18.5.1.1	18.5.1.3	18.4.1.1	18.4.1.4	18.3.1.1
00000000	00000000	0000000	0000000	000000	000000	0000
00011111	00111110	0001111	0001111	000111	001110	0011
01100221	0101220	0110221	0110111	0110101	010111	0101
01122110	0122011	0112110	0121210	011120	01220	0120
02212020	0212201	022120	021220	02210	02121	0210
02221201	0220121	022201	022201	02221	02201	0221
10111220	1002211	101100	101020	10110	10020	1001
10122001	1021021	102021	102210	10201	10211	1020
11201010	1112120	110201	110221	11021	11121	1110
11210101	1120200	112010	111101	11200	11200	1121
12002211	1201101	120220	120100	12020	12011	1200
12020120	1210010	121111	122011	12111	12100	1211
20202121	2010221	201221	201201	20121	20101	2010
20220210	2022100	202210	202121	20220	20221	2021
21012200	2100111	210120	210210	21010	21001	2100
21021021	2111001	212101	212000	21211	21110	2111
22101100	2202020	220011	220021	22001	22020	2201
22110011	2221210	221000	221110	22100	22210	2220

Table 8: The series of OA(24; $3^1 2^4; 3$)

24.1.4.1	24.1.4.2	24.1.4.3
00000	00000	00000
00001	00011	00011
00110	00101	00101
00111	00110	00110

continued on next page

Table 8 (continued)

24.1.4.1	24.1.4.2	24.1.4.3
01010	01001	01001
01011	01010	01010
01100	01100	01100
01101	01111	01111
10000	10000	10000
10011	10011	10011
10101	10101	10101
10110	10110	10110
11001	11001	11001
11010	11010	11010
11100	11100	11100
11111	11111	11111
20010	20000	20001
20011	20011	20010
20100	20101	20100
20101	20110	20111
21000	21001	21000
21001	21010	21011
21110	21100	21101
21111	21111	21110

Table 9: The set of OA(48; 3^{12^9} ; 3)

48.1.9.1	48.1.9.2	48.1.9.3
0000000000	0000000000	0000000000
0000001111	0000001111	0000001111
0000110011	0000110011	0000110011
0000111100	0000111100	0000111100
0011000011	0011000011	0011000100
0011001100	0011001100	0011010011
0011110000	0011110000	0011101001
0011111111	0011111111	0011111110
0101010101	0101010101	0101001010
0101011010	0101011010	0101011101
0101100110	0101100110	0101100111
0101101001	0101101001	0101110000
0110010110	0110010110	0110010110
0110011001	0110011001	0110011001
0110100101	0110100101	0110100101
0110101010	0110101010	0110101010
1000010101	1000010101	1000010110
1001000110	1001000110	1001000101
1001011000	1001011011	1001011011
1001100001	1001100001	1001101110
1010011110	1010011110	1010001000
1010100111	1010100100	1010100011

continued on next page

Table 9 (continued)

48.1.9.1	48.1.9.2	48.1.9.3
1010111001	1010111001	1010111101
1011101010	1011101010	1011110000
1100001011	1100001000	1100010001
1100101100	1100101111	1100100100
1100110010	1100110010	1100111010
1101111111	1101111100	1101101001
1110000000	1110000011	1110001111
1111001101	1111001101	1111000010
1111010011	1111010000	1111011100
1111110100	1111110111	1111110111
2000101010	2000101010	2000101001
2001011011	2001011000	2001011000
2001101101	2001101101	2001100010
2001110110	2001110110	2001110101
2010001001	2010001001	2010010101
2010010010	2010010010	2010011010
2010100100	2010100111	2010100110
2011010101	2011010101	2011001111
2100000111	2100000100	2100000011
2100011100	2100011111	2100001100
2100110001	2100110001	2100111111
2101000000	2101000011	2101010110
2110111111	2110111100	2110110000
2111001110	2111001110	2111000001
2111100011	2111100000	2111101100
2111111000	2111111011	2111111011

Table 10: The set of OA(54; $3^5 2^1$; 3)

54.5.1.1	54.5.1.2	54.5.1.3	54.5.1.4
000000	000000	000000	000000
000011	000011	000011	000011
001101	001101	001101	001101
001120	001120	001120	001120
002210	002210	002210	002210
002221	002221	002221	002221
010110	010110	010110	010110
010220	010221	010221	010221
011011	011011	011011	011011
011200	011200	011200	011200
012021	012020	012020	012020
012101	012101	012101	012101
020121	020121	020120	020121
020201	020200	020201	020200
021020	021020	021021	021020
021211	021211	021210	021211

continued on next page

Table 10 (continued)

54.5.1.1	54.5.1.2	54.5.1.3	54.5.1.4
0 2 2 0 0 0	0 2 2 0 0 1	0 2 2 0 0 0	0 2 2 0 0 1
0 2 2 1 1 0	0 2 2 1 1 0	0 2 2 1 1 1	0 2 2 1 1 0
1 0 0 1 1 1	1 0 0 1 1 1	1 0 0 1 1 1	1 0 0 1 1 1
1 0 0 2 2 1	1 0 0 2 2 0	1 0 0 2 2 0	1 0 0 2 2 0
1 0 1 0 1 0	1 0 1 0 1 0	1 0 1 0 1 0	1 0 1 0 1 0
1 0 1 2 0 1	1 0 1 2 0 1	1 0 1 2 0 1	1 0 1 2 0 1
1 0 2 0 2 0	1 0 2 0 2 1	1 0 2 0 2 1	1 0 2 0 2 1
1 0 2 1 0 0	1 0 2 1 0 0	1 0 2 1 0 0	1 0 2 1 0 0
1 1 0 0 0 1	1 1 0 0 0 1	1 1 0 0 0 1	1 1 0 0 0 1
1 1 0 0 2 0	1 1 0 0 2 0	1 1 0 1 0 0	1 1 0 1 0 0
1 1 1 1 1 0	1 1 1 1 1 0	1 1 1 1 2 1	1 1 1 1 2 1
1 1 1 1 2 1	1 1 1 1 2 1	1 1 1 2 2 0	1 1 1 2 2 0
1 1 2 2 0 0	1 1 2 2 0 0	1 1 2 0 1 0	1 1 2 0 1 0
1 1 2 2 1 1	1 1 2 2 1 1	1 1 2 2 1 1	1 1 2 2 1 1
1 2 0 1 0 0	1 2 0 1 0 0	1 2 0 0 2 0	1 2 0 0 2 1
1 2 0 2 1 0	1 2 0 2 1 1	1 2 0 2 1 1	1 2 0 2 1 0
1 2 1 0 0 1	1 2 1 0 0 1	1 2 1 0 0 1	1 2 1 0 0 0
1 2 1 2 2 0	1 2 1 2 2 0	1 2 1 1 1 0	1 2 1 1 1 1
1 2 2 0 1 1	1 2 2 0 1 0	1 2 2 1 2 1	1 2 2 1 2 0
1 2 2 1 2 1	1 2 2 1 2 1	1 2 2 2 0 0	1 2 2 2 0 1
2 0 0 1 2 0	2 0 0 1 2 0	2 0 0 1 2 1	2 0 0 1 2 0
2 0 0 2 0 0	2 0 0 2 0 1	2 0 0 2 0 0	2 0 0 2 0 1
2 0 1 0 2 1	2 0 1 0 2 1	2 0 1 0 2 0	2 0 1 0 2 1
2 0 1 2 1 0	2 0 1 2 1 0	2 0 1 2 1 1	2 0 1 2 1 0
2 0 2 0 0 1	2 0 2 0 0 0	2 0 2 0 0 1	2 0 2 0 0 0
2 0 2 1 1 1	2 0 2 1 1 1	2 0 2 1 1 0	2 0 2 1 1 1
2 1 0 1 0 1	2 1 0 1 0 1	2 1 0 0 2 1	2 1 0 0 2 0
2 1 0 2 1 1	2 1 0 2 1 0	2 1 0 2 1 0	2 1 0 2 1 1
2 1 1 0 0 0	2 1 1 0 0 0	2 1 1 0 0 0	2 1 1 0 0 1
2 1 1 2 2 1	2 1 1 2 2 1	2 1 1 1 1 1	2 1 1 1 1 0
2 1 2 0 1 0	2 1 2 0 1 1	2 1 2 1 2 0	2 1 2 1 2 1
2 1 2 1 2 0	2 1 2 1 2 0	2 1 2 2 0 1	2 1 2 2 0 0
2 2 0 0 1 0	2 2 0 0 1 0	2 2 0 0 1 0	2 2 0 0 1 0
2 2 0 0 2 1	2 2 0 0 2 1	2 2 0 1 0 1	2 2 0 1 0 1
2 2 1 1 0 0	2 2 1 1 0 0	2 2 1 1 0 0	2 2 1 1 0 0
2 2 1 1 1 1	2 2 1 1 1 1	2 2 1 2 2 1	2 2 1 2 2 1
2 2 2 2 0 1	2 2 2 2 0 1	2 2 2 0 1 1	2 2 2 0 1 1
2 2 2 2 2 0	2 2 2 2 2 0	2 2 2 2 2 0	2 2 2 2 2 0

References

- Block, R. and Mee, R. (2005), “Resolution IV Designs with 128 Runs,” *Journal of Quality Technology*, 37, 282-293.
- Box, G.E.P., Hunter, W.G., and Hunter, J.S. (2005), *Statistics for experimenters*, 2nd edition, New York: Wiley.
- Brouwer, A.E., Cohen, A.M., and Nguyen, M.V.M (2006), *Orthogonal arrays*

- of strength 3 and small run sizes, *Journal of Statistical Planning and Inference*, 136, 3268-3280.
- Chen, J., Sun, D.X., and Wu, C.F.J. (1993), "A Catalogue of Two-Level and Three-Level Fractional Factorial Designs with Small Runs," *International Statistical Review*, 61, 131-145.
- Cheng, C.-S., Steinberg, D.M., and Sun, D.X. (1999), "Minimum aberration and model robustness for two-level fractional factorial designs," *Journal of the Royal Statistical Society Series B*, 61, 85-93.
- Cheng, S.W., and Ye, K.Q. (2004), "Geometric isomorphism and minimum aberration for factorial designs with quantitative factors," *Annals of Statistics*, 32, 2168-2185.
- Clark, J.B., and Dean, A.M. (2001), "Equivalence of fractional factorial designs," *Statistica Sinica*, 11, 537-547.
- Connor, W.S. and S. Young (1961), "Fractional factorial designs for experiments with factors two or three levels," *U.S. Department of Commerce, National Bureau of Standards, Applied Mathematics Series*, 58.
- Deng, L.Y. and Tang, B. (2002), Design selection and classification for Hadamard matrices using generalized minimum aberration criterion," *Technometrics*, 44, 173-184.
- Dey, A., and Mukerjee, R. (1999) *Fractional factorial plans*, New York: Wiley.
- Evangelaras, H., Koukouvinos, C., Dean, A.M., and Dingus, C.A. (2005), "Projection properties of certain three-level orthogonal arrays," *Metrika*, 62, 241-257.
- Fries, A., and Hunter, W. G. (1980), "Minimum aberration $2k - p$ designs," *Technometrics*, 22, 601-608.
- Hedayat, A.S., Seiden, E., and Stufken, J. (1997), "On the maximal number of factors and the enumeration of 3-symbol orthogonal arrays of strength 3 and index 2," *Journal of Statistical Planning and Inference*, 58, 43-63.
- Hedayat, A.S., Sloane, N.J.A., and Stufken, J. (1999), *Orthogonal arrays: theory and applications*, New York: Springer.
- Jones, B.A., Li, W., Nachtsheim, C.J., and Ye, K.Q. (2007), "Model discrimination - another perspective on model-robust designs," *Journal of Statistical Planning and Inference*, 137, 1576-1583.
- Kaski, P., and Östergård, P.R.J. (2005), *Classification algorithms for codes and designs*, New York: Springer.
- Lenth, R.V. (1989), "Quick and easy analysis of unreplicated factorials," *Technometrics*, 31, 469-473.
- Li, W. (2006), "Screening designs for model selection," in *Screening: methods for experimentation in industry, drug discovery, and genetics.*, eds. A. Dean and S. Lewis, New York: Springer Science+Business Media.
- Loeppky, J.L., and Sitter, R.R. (2002), "Analyzing unreplicated blocked or split-plot fractional factorial designs," *Journal of Quality Technology*, 34, 229-243.
- Rao, C.R. (1947), "Factorial experiments derivable from combinatorial arrangements of arrays," *Journal of the Royal Statistical Society Supplement*, 9, 128-139.
- Schoen, E.D. (2007), "Classification of all orthogonal arrays with run-size 18," Research paper D/2007/1169/011, Faculty of Applied Economic, University of Antwerp, Belgium.

- Schoen, E.D., and Kaul, E.A.A. (2000), "Three robust scale estimators to judge unreplicated experiments," *Journal of Quality Technology*, 32, 276-283.
- Sitter, R.R., Chen, J., and Feder, M. (1997), "Fractional resolution and minimum aberration in blocked 2^{n-k} designs," *Technometrics*, 39, 382-390.
- Srivastava, J.N. (1975), "Designs for searching non-negligible effects," in *A survey of statistical design and linear models*, ed. J.N. Srivastava, Amsterdam: North-Holland, pp.507-519.
- Sun, D.X., Li, W., and Ye, K.Q. (2002), "An algorithm for sequentially constructing non-isomorphic orthogonal designs and its applications," preprint.
- Sun, D.X., Wu, C.F.J., and Chen, Y. (1997), "Optimal blocking schemes for 2^n and 2^{n-p} designs," *Technometrics*, 39, 298-307.
- Tsai, P.W., Gilmour, S.G., and Mead, R. (2000), "Projective three-level main effects designs robust to model uncertainty," *Biometrika*, 87, 467-475.
- Wu, C.F.J., and Hamada, M.S. (2000), *Experiments: Planning, Analysis, and Parameter Design Optimization*, New York: Wiley.
- Xu, H. (2002), "An algorithm for constructing orthogonal and nearly-orthogonal arrays with mixed levels and small runs," *Technometrics*, 44, 356-368.
- Xu, H. (2005), "A catalogue of three-level regular fractional factorial designs," *Metrika*, 62, 259-281.
- Xu, H., and Wu, C.F.J. (2001), "Generalized minimum aberration for asymmetrical fractional factorial designs," *The Annals of Statistics*, 29, 1066-1077.
- Xu, H., Cheng, S., and Wu, C.F.J. (2004) Optimal projective three-level designs for factor screening and interaction detection. *Technometrics*, 46, 280-292.