Algorithmic Construction of Efficient Fractional Factorial Designs With Large Run Sizes

Hongquan Xu

Department of Statistics

University of California

Los Angeles, CA 90095-1554

(hqxu@stat.ucla.edu)

February 5, 2009

Fractional factorial designs are widely used in practice and typically chosen according to the minimum aberration criterion. A sequential algorithm is developed for constructing efficient fractional factorial designs. A construction procedure is proposed that only allows a design to be constructed from its minimum aberration projection in the sequential build-up process. To efficiently identify nonisomorphic designs, designs are divided into different categories according to their moment projection patterns. A fast isomorphism check procedure is developed by matching the factors using their delete-one-factor projections. This algorithm is used to completely enumerate all 128-run designs of resolution 4, all 256-run designs of resolution 4 up to 17 factors, all 512-run designs of resolution 5, all 1024-run designs of resolution 6, and all 2048- and 4096-run designs of resolution 7. A method is proposed for constructing minimum aberration designs using only a partial catalog of some good designs. Three approaches are further suggested for constructing good designs with a large number of factors. Efficient designs, often with minimum aberration, are tabulated up to 40, 80, 160, 45, 47, and 65 factors for 128, 256, 512, 1024, 2048, and 4096 runs, respectively.

KEY WORDS: Fractional factorial design, isomorphism, linear code, MacWilliams identity, minimum aberration, resolution

1 Introduction

Fractional factorial (FF) designs are widely used in many areas of science, engineering and industry. With the rapidly increasing computational power, more and more large FF designs are used in large scale computer experiments where physical processes are being simulated. Lin and Sitter (2008) reported that FF designs with over 600 runs and as many as 53 parameters were used in computer simulations at Los Alamos National Laboratory. Kleijnen et al. (2005) reported a few computer simulations that investigated dozens or hundreds of factors.

Two-level FF designs with several hundred or thousand runs can be very useful in real applications. Consider an application described by Mee (2004) and Telford (2007). The researchers at Johns Hopkins University employed several two-level FF designs in a ballistic missile defense project to assess the sensitivity of 47 parameters of an extended air defense simulation in two far-term scenarios over the first 10 days of a war. In the first scenario, a resolution IV design with 512 runs was initially used and followed by 17 additional designs (for a total of 352 additional runs) to resolve aliasing of two-factor interactions. In the second scenario, the researchers used a resolution V design with 4096 runs constructed by SAS PROC FACTEX. Half of the 4096 runs could have been saved if they had obtained a resolution V design with 2048 runs; see Section 4 for such a design.

FF designs are often chosen by the minimum aberration (MA) criterion (Fries and Hunter 1980), an extension of the maximum resolution criterion (Box and Hunter 1961). Most textbooks and references in the literature provide MA designs up to 128 runs only; see, among others, Box, Hunter, and Hunter (2005), Dean and Voss (1999), Montgomery (2005), Mukerjee and Wu (2006), and Wu and Hamada (2000). The construction of efficient designs is very challenging when the run size is large. Few algorithms are available and they are not effective.

It was four decades ago when Draper and Mitchell first attacked this challenging problem seriously. Draper and Mitchell (1967, 1968) developed a stage-by-stage algorithm and completely enumerated all 256-run designs of resolution ≥ 5 and all even 512-run designs of resolution ≥ 6. An even design contains entirely defining words of even length whereas an odd design has at least one defining word of odd length. Draper and Mitchell (1970) attempted but failed to construct the complete set of even 1024-run designs of resolution ≥ 6 and the complete set of odd 512-run designs of resolution ≥ 5. They obtained 4,043 distinct even 1024-run designs of resolution ≥ 6; as we will see later, they missed about 30% designs.

The construction of efficient FF designs is relatively easier when the run size is smaller. Chen, Sun and Wu (1993, CSW hereafter) developed a sequential algorithm and enumerated all 8, 16, 27, 32-run designs of resolution ≥ 3 and 64-run designs of resolution ≥ 4. Xu (2005) extended their work and enumerated all 81-run designs of resolution ≥ 3, 243-run designs of resolution ≥ 4, and 729-run designs of resolution ≥ 5. Based on a conjecture, Block and Mee (2005) constructed MA 128-run designs for 12 to 64 factors. Lin and Sitter (2008) developed an algorithm and enumerated all 128-run designs of resolution ≥ 4 up to 16 factors, all 512-run designs of resolution ≥ 5 up to 17 factors, and all even 1024-run designs of resolution ≥ 6 up to 18 factors.

A key step in any algorithmic construction of FF designs is to determine whether two designs are isomorphic. Two FF designs are *isomorphic* (or *equivalent*) if and only if one may be obtained from the other by relabeling the factors and/or relabeling the levels of one or more factors. Two designs are distinct if they are not equivalent. For large FF designs, the test of equivalence of two designs requires an excessive amount of computer time, so many test procedures have been proposed to quickly identify nonisomorphic designs. Draper and Mitchell (1967) used the wordlength pattern to distinguish designs. Unfortunately, two nonisomorphic designs can have the same wordlength pattern, so Draper and Mitchell (1970) used a "letter pattern comparison" to test the equivalency of two designs and conjectured that FF designs with the same letter pattern are isomorphic. However, Chen and Lin (1991) disproved their conjecture by constructing two nonisomorphic 2^{31-15} designs with the same letter pattern. Zhu and Zeng (2005) reported that counter examples exist for as small as 32 runs; they also proposed a more sensitive test based on the coset pattern, which still fails to determine a design uniquely. Block and Mee (2005) conjectured that two designs are isomorphic if their sets of delete-one-factor projections are equivalent. See Clark and Dean (2001), Ma, Fang, and Lin (2001), Xu (2005), and Lin and Sitter (2008) for other test procedures.

In this paper we develop a new algorithm for constructing efficient FF designs with large run sizes. As in other algorithms, we construct designs sequentially by adding one column at a time. We introduce an intelligent construction procedure that only allows a design to be constructed from its MA projection in the sequential build-up process. This procedure discards many isomorphic designs without performing time-consuming isomorphism checks. As we will see later, this procedure is more efficient than the procedure used by Lin and Sitter (2008) who adopted a combined approach from Bingham and Sitter (1999). To identify nonisomorphic designs, we divide designs into different categories according to their moment projection patterns. As demonstrated by Xu (2005), the use of moment projection patterns is more efficient than the use of letter patterns in terms of both distinguishing designs and computation. To test whether two designs in the same category are isomorphic, we develop a fast isomorphism check procedure by matching the factors using their delete-one-factor projections. This procedure skips many unsuccessful relabeling maps and is much more efficient than the procedures used by CSW and Lin and Sitter (2008). The new algorithm enables us to completely enumerate all 128-run designs of resolution ≥ 4, all 256-run designs of resolution ≥ 4 up to 17 factors, all 512-run designs of resolution ≥ 5, all 1024-run designs of resolution ≥ 6, and all 2048- and 4096-run designs of resolution ≥ 7. Based on an upper bound on the wordlength pattern, we propose a method for constructing MA designs using only a partial catalog of some good designs. This enables us to construct MA designs efficiently when the run size or the number of factors is small. However, as both the run size and the number of factors increase, the construction of MA designs becomes infeasible thus we further propose three approaches for constructing good designs. We tabulate efficient designs up to 40, 80, 160, 45, 47, and 65 factors for 128, 256, 512, 1024, 2048, and 4096 runs, respectively. For clarity, we consider only two-level *regular* FF designs. The extension to multi-level designs is straightforward.

In Section 2, we review some basic concepts, definitions and preliminary results. We describe the construction methods in Section 3. Tables of designs with 128–4096 runs are given in Section 4 and concluding remarks are given in Section 5.

2 Basic concepts, definitions and preliminary results

A regular 2^{n-k} FF design, denoted by D, has n factors of two levels and 2^{n-k} runs. A factor is also called a letter or a column whereas a run is called a row. Associated with every regular 2^{n-k} design is a set of k independent defining words. The defining contrast subgroup of D consists of all possible products of the k defining words and has 2^k words (including the identity I). Let $A_i(D)$ be the number of words of length i. The vector $(A_1(D), \ldots, A_n(D))$ is called the *wordlength pattern*. The *resolution* is the smallest i such that $A_i(D) > 0$.

Let D_1 and D_2 be two regular 2^{n-k} designs. D_1 is said to have less aberration than D_2 if there exists an r such that $A_i(D_1) = A_i(D_2)$ for $i = 1, \ldots, r-1$ and $A_r(D_1) < A_r(D_2)$. D_1 is said to have *minimum aberration* (MA) if there is no other regular design with less aberration than D_1.

A 2^{n-k} design D of resolution R is said to have *weak MA* (Chen and Hedayat 1996) if it has maximum resolution and $A_R(D)$ is minimized among all regular designs.

2.1 Connection with coding theory

The connection between factorial designs and linear codes is important in the development of our algorithm. For an introduction to coding theory, see Hedayat, Sloane, and Stufken (1999, Chapter 4) and MacWilliams and Sloane (1977).

A regular 2^{n-k} FF design D is also known as a linear code of length n and dimension $n-k$ over the binary field $GF(2)$ in coding theory. Associated with every binary linear code is another linear code, the dual code $D^{\perp}$, that consists of all row vectors $(u_1, \ldots, u_n)$ over $GF(2)$ such that $\sum_{i=1}^{n} u_i v_i = 0$ for all $(v_1, \ldots, v_n)$ in D.

The *Hamming weight* of a vector $(u_1, \ldots, u_n)$ is the number of nonzero components u_i. Let $B_i(D)$ and $B_i(D^{\perp})$ be the number of rows with Hamming weight i in D and $D^{\perp}$, respectively. The vectors $(B_0(D), B_1(D), \ldots, B_n(D))$ and $(B_0(D^{\perp}), B_1(D^{\perp}), \ldots, B_n(D^{\perp}))$ are called the *weight distributions* of D and $D^{\perp}$.

The weight distributions of D and $D^{\perp}$ are related through the *MacWilliams identities*.

$$B_j(D^{\perp}) \quad = \quad 2^{-(n-k)} \sum_{i=0}^{n} P_j(i;n) B_i(D) \text{ for } j = 0, \ldots, n, \tag{1}$$

where $P_j(x;n) = \sum_{i=0}^{j} (-1)^i \binom{x}{i} \binom{n-x}{j-i}$ are the *Krawtchouk polynomials*.

It is easy to see from the definitions that the defining contrast subgroup of D is indeed the dual code $D^{\perp}$ and that the wordlength pattern of D is the weight distribution of $D^{\perp}$, that is,

$$A_i(D) = B_i(D^{\perp}) \text{ for } i = 1, \ldots, n.$$

By definition, the wordlength pattern is computed via counting words in the defining contrast subgroup. This direct approach can be cumbersome when k is large, because there are 2^k words in a 2^{n-k} design. The connection with coding theory leads to an alternative approach. We can compute $A_i(D)$ via the weight distribution $B_i(D)$ and the MacWilliams identities (1). The Krawtchouk polynomials need to be computed once for each n and can be efficiently calculated via the following recursive identity:

$$P_j(x;n) = P_j(x-1;n) - P_{j-1}(x;n) - P_{j-1}(x-1;n)$$

and the initial values $P_0(x;n) = 1$ and $P_j(0;n) = \binom{n}{j}$. We use the alternative approach in our algorithm, because it is faster than the direct approach when $k > n-k$.

2.2 Delete-one-factor projections

For a 2^{n-k} design D and $i = 1, \ldots, n$, let $D(-i)$ be the resulting $2^{(n-1)-(k-1)}$ design when the ith column is deleted. These sub-designs are called the delete-one-factor projections of D. Note that $D(-i)$ may be degenerate in the sense that it has less than 2^{n-k} distinct runs.

The next two properties about MA delete-one-factor projections are important in our construction.

Lemma 1. *For a 2^{n-k} design D, if $D(-i)$ has MA among all delete-one-factor projections of D, then the ith column is a product of some of the other columns and therefore $D(-i)$ is not degenerate.*

Proof. Suppose the result is not true, then the ith column is independent of the other columns and therefore it does not appear in any word of D. Then we can choose another column that appears in some word and deleting that column would yield a design having less aberration than $D(-i)$, which is a contradiction. □

Lemma 2. *Suppose that D is a 2^{n-k} design of resolution R with δ_n words of length R. If $D(-i)$ has MA among all delete-one-factor projections of D, then $D(-i)$ has at most $\delta_n - \lceil R \cdot \delta_n/n \rceil$ words of length R, where $\lceil x \rceil$ is the smallest integer that is greater than or equal to x.*

Proof. Each word of length R consists of R factors, so on average each factor appears in $R \cdot \delta_n/n$ words of length R. There must exist a factor that appears in at least $\lceil R \cdot \delta_n/n \rceil$ words. Deleting this factor yields a design that has at most $\delta_n - \lceil R \cdot \delta_n/n \rceil$ words of length R. The lemma follows from the fact that MA projection $D(-i)$ has the least number of words of length R. □

3 Construction Methods

3.1 Basic idea

Following CSW, we construct designs sequentially by adding one factor at a time. We first review the basic idea of CSW's algorithm and then describe how to improve it.

Denote $r = n - k$. Let $\mathbf{G}$ be an $r \times (2^r - 1)$ matrix that consists of all nonzero r-tuples $(u_1, \ldots, u_r)^T$ from $GF(2)$. It is well known that every regular 2^{n-k} FF design can be viewed as n columns of an $2^r \times (2^r - 1)$ matrix $\mathbf{H}$, which consists of all linear combinations of the rows of $\mathbf{G}$ over $GF(2)$.

Let $C^R_{n,k}$ be the set of nonisomorphic 2^{n-k} designs of resolution $\geq R$. CSW constructed $C^R_{n+1,k+1}$ from $C^R_{n,k}$ by adding an additional column. For each design in $C^R_{n,k}$, there are $2^r - 1 - n$ ways to add a column to produce a design with $n+1$ columns. Let $\tilde{C}_{n+1,k+1}$ be the set of these designs. Obviously, $|\tilde{C}_{n+1,k+1}| = (2^r - 1 - n)|C^R_{n,k}|$. It is evident that $C^R_{n+1,k+1}$ is a subset of $\tilde{C}_{n+1,k+1}$. However, some designs in $\tilde{C}_{n+1,k+1}$ are isomorphic and some may have resolutions less than R. To construct $C^R_{n+1,k+1}$, it is necessary to eliminate these redundant designs. It is easy to eliminate designs of resolution $< R$ but is more difficult to eliminate isomorphic designs. To speed up the isomorphism check process, CSW divided all designs into different categories according to their wordlength patterns and letter patterns. Obviously, designs in different categories are not isomorphic. However, designs in the same category are not necessarily isomorphic and therefore a complete isomorphism check has to be applied to determine whether or not two designs are isomorphic.

3.2 A modified construction procedure

One problem with CSW's algorithm is that too many isomorphic designs are generated in the sequential build-up process, because a $2^{(n+1)-(k+1)}$ design can be generated from as many as $n+1$ distinct 2^{n-k} designs. We solve this problem by only allowing a design to be generated from its MA delete-one-factor projection.

We modify the construction procedure as follows. For any design D in $C^R_{n,k}$, adding a column to D yields a candidate design D_c. Discard D_c if its resolution is less than R or if D does not have MA among all delete-one-factor projections of D_c.

For illustration consider the construction of 2^{7-3} designs. According to CSW, there are four distinct 2^{6-2} designs and five distinct 2^{7-3} designs, labeled as 6-2.i and 7-2.j, where the designs are ranked according to the MA criterion. For each 2^{6-2} design, we can add one of the remaining 9 columns to obtain a 2^{7-2} design. Table 1 shows the number of times that each 2^{7-3} design is generated in the (unmodified) sequential construction. For example, design 7-3.3 is generated three times from design 6-2.2, nine times from design 6-2.3 and four times from design 6-2.4. The modified construction procedure only allows design 7-3.3 to be generated from design 6-2.2, because it has MA among all delete-one-factor projections of design 7-3.3. Under the original construction procedure we need to entertain $4 \times 9 = 36$ designs whereas under the modified construction procedure we need to entertain only 14 designs (boldfaced in Table 1). Because there are five distinct 2^{7-3} designs, we reduce the number of isomorphism checks from 31 to 9.

Table 1: Number of Times that 2^{7-3} Designs are Generated in the Sequential Construction

		2^{7-3} Designs				
Design	A_3	7-3.1	7-3.2	7-3.3	7-3.4	7-3.5
6-2.1	0	**2**	**6**	0	**1**	0
6-2.2	1	0	6	**3**	0	0
6-2.3	2	0	0	9	0	0
6-2.4	2	0	2	4	1	**2**

Table 2: Number of Designs Entertained in Creating Catalogs of 128-run Designs of Resolution ≥ 4

	n								
Procedure	8	9	10	11	12	13	14	15	16
CSW	99	458	1,104	2,597	6,632	16,200	36,192	79,064	160,040
Bingham and Sitter	99	186	506	1,367	3,499	7,950	15,798	29,062	48,889
Author	99	299	341	502	890	1,952	4,028	7,969	14,176
True	5	13	33	92	249	623	1,535	3,522	7,500

Bingham and Sitter (1999) proposed a construction procedure that combines the search table method of Franklin and Bailey (1977) and Franklin (1985) with the sequential approach. Table 2 shows the comparison of the construction procedures in the construction of 128-run designs of resolution ≥ 4. The last row of the table shows the number of distinct designs. As the table shows, both the combined procedure of Bingham and Sitter (1999) and our modified procedure significantly reduce the number of designs entertained. For $n \geq 10$, our modified procedure entertains substantially fewer designs than the other two procedures.

We now show, by induction, that every possible 2^{n-k} design of resolution $\geq R$ in 2^r runs is isomorphic to a design in $C^R_{n,k}$ under the modified construction procedure. It is trivial that this is true for $n = r + 1$. Suppose this is true for $n = r + k$. Consider $n + 1 = r + k + 1$. Let $D = (c_1, \ldots, c_{n+1})$ be a $2^{(n+1)-(k+1)}$ design of resolution $\geq R$ in 2^r runs. Suppose that $D(-i)$ has MA among all possible delete-one-factor projections of D. Lemma 1 implies that $D(-i)$ must be a non-degenerate 2^{n-k} design of resolution $\geq R$. By the assumption for 2^{n-k} designs, there exists a design D_n in $C^R_{n,k}$ that is isomorphic to $D(-i)$. Let π be the isomorphic map from $D(-i)$ to D_n, i.e., $D_n = \pi(D(-i))$. Note that $\pi(c_i)$ is uniquely defined under this isomorphic map.

Let $\pi(D) = (D_n, \pi(c_i))$. Clearly $\pi(D)$ is entertained in the modified construction procedure and therefore D is isomorphic to a design in $C^R_{n+1,k+1}$. This completes the proof.

3.3 A nonisomorphism classification procedure

Xu (2005) observed that the use of wordlength patterns and letter patterns is not efficient in identifying nonisomorphic designs for three-level FF designs. Following Xu (2005), we divide designs into different categories according to their weight distributions and moment projection patterns (to be defined next). As explained in Section 2.1, the use of weight distributions is equivalent to the use of wordlength patterns in terms of distinguishing designs but is more efficient in terms of computation (when $k > r$).

For a 2^{n-k} design D and an integer p, $p < n$, there are $\binom{n}{p}$ p-factor projections. For each p-factor projection, say D_p, and an integer t, compute the tth power moment

$$K_t(D_p) = \sum_{i=0}^{p} (p-i)^t B_i(D_p),$$

where $B_i(D_p)$ is the number of row vectors of D_p with Hamming weight i. The power moment K_t was introduced by Xu (2003) and Xu and Deng (2005) for ranking and classifying nonregular designs. The frequency distribution of K_t-values of all p-factor projections is called the *p-dimensional K_t-value distribution*. It is evident that isomorphic designs have the same p-dimensional K_t-value distribution for all positive integers t and $p < n$. Whenever two designs have different p-dimensional K_t-value distributions for some t and p, these two designs must be nonisomorphic.

To ease the computation, we fix t and let p vary from $n-1$ to $n-q$, where q is a pre-chosen small number, say 2 or 3. The corresponding q K_t-value distributions are called the moment projection pattern. It requires $O(n^q)$ operations to compute the moment projection pattern. The choice of t does not make a difference provided $t > 5$ in most cases. In the algorithm, we fix t arbitrarily at $t = 10$.

Table 3 shows the numbers of designs identified by the wordlength pattern (WLP), letter pattern (LP), moment projection pattern (MPP) with $q = 1$ and 2 in the construction of 128-run designs of resolution ≥ 4 for $n \leq 16$. Note that the moment projection pattern with $q = 1$ and the letter pattern identify the same numbers of designs. The moment projection pattern with $q = 2$ correctly identifies all nonisomorphic designs for $n \leq 16$.

As Table 3 shows, the moment projection pattern check with $q = 1$ has the same or nearly the same classification power as the letter patten check whereas the moment projection pattern check

Table 3: Number of Designs Identified for 128-Run Designs of Resolution ≥ 4

	n								
Method	8	9	10	11	12	13	14	15	16
WLP	5	13	28	68	152	297	518	889	1,425
LP	5	13	33	92	247	617	1,506	3,467	7,229
MPP ($q = 1$)	5	13	33	92	247	617	1,506	3,467	7,229
MPP ($q = 2$)	5	13	33	92	249	623	1,535	3,522	7,500

with $q = 2$ or 3 typically has more classification power. Furthermore, when k is large, the moment projection pattern check is faster than the letter pattern check.

3.4 A fast isomorphism check procedure

We first review the isomorphism check procedure proposed by CSW. Consider two 2^{7-3} designs defined by

$$D_1 : 5 = 123, 6 = 124, 7 = 13 \text{ and } D_2 : 5 = 12, 6 = 124, 7 = 234,$$

which have the same wordlength pattern and letter pattern. CSW's procedure works as follows:

1. Select four independent columns from D_2, say, $\{1, 2, 3, 6\}$. There are $\binom{7}{4}$ choices.
2. Select a relabeling map from $\{1, 2, 3, 6\}$ to $\{a, b, c, d\}$, say, $a = 1$, $b = 2$, $c = 3$, and $d = 6$. There are 4! choices.
3. Write the remaining columns, $\{4, 5, 7\}$, in D_2 as interactions of $\{a, b, c, d\}$, i.e., $4 = abd$, $5 = ab$, and $7 = acd$. Then D_2 can be written as $\{a, b, c, d, ab, abd, acd\}$.
4. Compare the new representation of D_2 with that of D_1. If they match, D_1 and D_2 are isomorphic, and the process stops. Otherwise, return to step 2 and try another map of $\{a, b, c, d\}$. When all the relabeling maps are exhausted, return to step 1 and find next four columns.

If two designs are isomorphic, an isomorphic map will be found eventually. Otherwise, two designs are not isomorphic. In the worst case, it requires $O(n\binom{n}{r}r!)$ operations to declare that two 2^{n-k} designs are not isomorphic.

Table 4: Weight Distributions of Delete-One-Factor Projections

D_1								D_2							
Projection	B_0	B_1	B_2	B_3	B_4	B_5	B_6	Projection	B_0	B_1	B_2	B_3	B_4	B_5	B_6
$D_1(-1)$	1	0	4	6	3	2	0	$D_2(-1)$	1	0	4	6	3	2	0
$D_1(-2)$	1	0	4	6	3	2	0	$D_2(-2)$	1	0	4	6	3	2	0
$D_1(-3)$	1	0	4	6	3	2	0	$D_2(-3)$	1	1	2	6	5	1	0
$D_1(-4)$	1	1	2	6	5	1	0	$D_2(-4)$	1	0	4	6	3	2	0
$D_1(-5)$	1	0	4	6	3	2	0	$D_2(-5)$	1	0	3	8	3	0	1
$D_1(-6)$	1	1	2	6	5	1	0	$D_2(-6)$	1	0	4	6	3	2	0
$D_1(-7)$	1	0	3	8	3	0	1	$D_2(-7)$	1	1	2	6	5	1	0

We improve the isomorphism check procedure by considering delete-one-factor projections. Let π be a permutation of $\{1, \ldots, n\}$. If π is an isomorphic map from D_1 to D_2, $D_1(-i)$ and $D_2(-\pi(i))$ must be isomorphic and therefore they must have the same weight distribution. So π cannot be an isomorphic map if $D_1(-i)$ and $D_2(-\pi(i))$ do not have the same weight distribution for some i.

For convenience, we call a permutation π *feasible* if $D_1(-i)$ and $D_2(-\pi(i))$ have the same weight distribution for every i. A relabeling map is feasible if its induced permutation is feasible. The key idea of our new isomorphism check procedure is to entertain only feasible relabeling maps by matching the factors using the weight distributions of the delete-one-factor projections.

We illustrate our procedure with the two 2^{7-3} designs mentioned earlier. Here are the steps.

1. Compute the weight distributions of the delete-one-factor projections (delete-one weight distributions, for short) for both designs; see Table 4. For each column of D_1, count the frequency that each delete-one weight distribution appears. Let n_i be the frequency for the ith column. Here $n_1 = n_2 = n_3 = n_5 = 4$, $n_4 = n_6 = 2$ and $n_7 = 1$.

2. Relabel the columns of D_1 by selecting four new independent columns so that their frequency numbers n_i are as small as possible. For example, we select columns $\{7, 4, 6, 1\}$ as the new independent columns. We relabel them as $\{a, b, c, d\}$, i.e., $a = 7, b = 4, c = 6, d = 1$, and write the remaining three columns as their interactions, i.e., $2 = bcd$, $3 = ad$, and $5 = abcd$. So after relabeling, D_1 becomes $D_1' : \{a, b, c, d, ad, bcd, abcd\}$. The purpose of this step is to reduce the number of feasible relabeling maps to be considered in the next step.

3. Select four independent columns from D_2 that have the same delete-one weight distributions

Table 5: Time to Create Catalogs of 128-Run Designs of Resolution ≥ 4

	n								
Algorithm	8	9	10	11	12	13	14	15	16
CSW	0s	1s	4s	27s	2m32s	10m30s	37m48s	2h27m	6h43m
Author	0s	0s	0s	1s	1s	4s	8s	16s	39s

NOTE: The CSW's algorithm is modified so that two algorithms differ only in the isomorphism check procedures used. The h, m, and s stand for hour, minute, and second, respectively.

as the four independent columns from D_1', and relabel the columns. To obtain a feasible map from D_2 to D_1', we must relabel column 5 of D_2 as a, because only column 5 has the same delete-one weight distribution as factor a of D_1'. Similarly, we must relabel column 3 or 7 of D_2 as b or c. We can relabel column 1, 2, 4, or 6 of D_2 as d. There are $1 \times 2 \times 4 = 8$ choices of feasible relabeling maps. For example, we choose $a = 5, b = 3, c = 7, d = 1$ and write the remaining columns as $2 = ad, 4 = abcd, 6 = bcd$. It is clear now that D_2 is isomorphic to D_1' and hence to D_1.

4. If two designs do not match after relabeling the independent columns, consider another choice of relabeling and/or another choice of independent columns in step 3. If none of the choices yields to an identical design, two designs are not isomorphic.

In the above example, we entertain only eight feasible relabeling maps out of $\binom{7}{4}4! = 840$ possible choices of relabeling maps. It can be verified that any of the eight feasible relabeling maps leads to an isomorphic map. This is not true in general.

In theory our new isomorphism check procedure still requires $O(n\binom{n}{r}r!)$ operations in the worst case. In practice, the new isomorphism check procedure saves tremendous amount of computing time, because the worst case happens rarely.

To see the computational advantage of the our new isomorphism check procedure, we develop two algorithms with everything the same except isomorphism check procedures, one with the original procedure by CSW and the other with our new procedure. Table 5 shows the real time comparison of these two procedures in constructing 128-run designs of resolution ≥ 4. The savings are tremendous and become larger for larger designs. The times are taken on a 2GHz PowerPC G5 computer.

Table 6: Illustration of Constructing MA 256-Run Designs for $n \leq 28$

n	9	10	11	12	13	14	15	16	17	18
δ_n	0	0	0	0	0	1	2	3	5	7
$\lvert C^4_{n,k}(\delta_n)\rvert$	5	9	11	14	15	124	617	1,836	14,158	46,929
n	19	20	21	22	23	24	25	26	27	28
δ_n	9	12	16	20	25	31	38	46	54	64
$\lvert C^4_{n,k}(\delta_n)\rvert$	56,821	104,654	258,535	136,105	65,070	23,981	5,610	661	6	1

The isomorphism check can be made faster in some situations. It is evident that two designs are isomorphic if and only if their dual codes are isomorphic. So when $k < r$, we perform isomorphism checks on the dual codes. This technique was previously used by Lin and Sitter (2008).

As an alternative, we can match columns using their letter patterns. It can be shown that the use of delete-one weight distributions is equivalent to the use of letter patterns. We use the former because it is faster to compute delete-one weight distributions than letter patterns when $k > r$.

Clark and Dean (2001) presented a method of determining isomorphism of any two FF designs, regular or nonregular, by examining the Hamming distances of their projection designs. They also developed an algorithm for checking the isomorphism of two-level designs. Their isomorphism check procedure, adopted by Lin and Sitter (2008), is inferior to ours for the regular design case, because it ignores the special properties of regular designs and requires $O(n(n!)^2)$ operations in theory for the worst case.

3.5 Construction of MA designs

It is infeasible to enumerate all designs in many situations. Here we propose a method for constructing MA designs by enumerating a subset of good designs.

Let $C^R_{n,k}(\delta_n)$ be the set of nonisomorphic 2^{n-k} designs of resolution $\geq R$ with at most δ_n words of length R. We can sequentially build up $C^R_{n,k}(\delta_n)$ as before. To construct $C^R_{n,k}(\delta_n)$, according to Lemma 2, it is sufficient to add a column to every design in $C^R_{n-1,k-1}(\delta_{n-1})$, where

$$\delta_{n-1} = \delta_n - \left\lceil \frac{R \cdot \delta_n}{n} \right\rceil . \tag{2}$$

For illustration, consider the construction of MA 256-run designs for $n \leq 28$. It is known from Block (2003) that there is a resolution IV 2^{28-20} design with $A_4 = 64$. We set $R = 4$, $\delta_{28} = 64$, and compute δ_{n-1} backward using (2) recursively for $n = 28, \ldots, 10$. Then we build up $C^4_{n,k}(\delta_n)$ forward for $n = 9, \ldots, 28$. By completely enumerating $C^4_{n,k}(\delta_n)$, we obtain all MA 256-run designs for $n \leq 28$. Table 6 shows the value of δ_n and the cardinality of $C^4_{n,k}(\delta_n)$. From the table, we know that there is a unique resolution IV 2^{28-20} design with $A_4 \leq 64$. The 14,158 designs that must be considered at $n = 17$ (to verify that the 28-factor design has MA) represent fewer than 1% of the resolution IV designs.

As the example shows, the method is very effective in reducing the number of designs to be evaluated in the construction of MA designs. It works well when the run size or the number of factors is small, or as long as we can enumerate all the distinct designs encountered at each stage. However, this becomes infeasible when both the run size and the number of factors are large, simply because there are too many designs to be enumerated. Indeed, we fail to construct MA 256-run designs for $n \geq 29$ because we encounter several million designs which exhaust the computer memory. The construction of MA designs is extremely difficult, if not infeasible, for larger runs and larger n.

3.6 Construction of good designs

Good designs with dozens of factors and several hundred or thousand runs are also useful in real applications but require further effort to obtain them. Here we propose three approaches, similar to what Block (2003) used in the construction of 256-run designs.

The first approach is a simple modification of the basic algorithm. We limit the number of distinct designs retained in the sequential search. Specially, we sort the designs according to the MA criterion, then build up from them to at most M designs at each stage. To speed up the search, we often skip the isomorphism check and distinguish designs using wordlength patterns only. This simple modification works well for small n when the basic algorithm fails. Indeed, most of the MA designs can be quickly obtained in this way. Two shortcomings of this approach are (i) there may be no eligible designs at some stage and (ii) the resulting designs may depend on those retained in the previous stages. To alleviate these shortcomings, we randomize the order of the columns to be added in the sequential build-up process and run the algorithm a few times, which may lead to some improved designs.

The second approach is to perform a random stepwise search and maintain a list of best designs during the search. We randomly generate designs by adding one column at a time. At each stage we retain only one design, compare it with the best design in the list using the MA criterion and

Table 7: Methods Used in the Construction of Good Designs

	Run Size					
Method	128	256	512	1024	2048	4096
Basic Algorithm	≤ 40	≤ 28	≤ 25	≤ 24	≤ 23	≤ 24
Approach 1		29–40	26–50	25–31 & 34–45	35–40	25–30
Approach 2		41–80	51–98		24–31	31–41
Approach 3			99–160	32–33	32–34 & 41–47	42–65

update the list if it is better. If a better design is found, we further perform a naive backward search which compares its MA delete-one-factor projection with the best design in the list. We repeat the naive backward search until no better projection designs can be found. The random search, only involving the computation and comparison of wordlength patterns, is very fast; therefore, we can repeat the whole process a large number of times, say L times. This approach can construct some good designs with large n when the first approach fails. For example, we obtain MA 256-run designs with n=69–80 via this approach; see next section for more details.

The third approach is to start with some known good designs with large n and perform a naive backward search. The doubling method proposed by Chen and Cheng (2006) can be used to construct good resolution IV designs with large n. Specially, by repeatedly doubling the 2^{5-1} design defined by $I = ABCDE$, we can construct a resolution IV design with 16×2^k runs and 5×2^k factors for any $k \geq 1$. Chen and Cheng (2006) showed that such a design has MA and its projection designs are also good when n is close to 5×2^k. For resolution V designs, the doubling method does not work. Since a 2^{n-k} design of resolution V is equivalent to a binary linear code with length n, dimension k and minimum distance 5, we can use some existing linear codes. In particular, we obtain 2^{33-23}, 2^{47-36} and 2^{65-53} designs of resolution V from the corresponding linear codes in Chen (1991) and Brouwer (1998). By folding over the first two designs, we further obtain 2^{34-23} and 2^{48-36} designs of resolution VI. This approach gives us a few more good designs with large n.

When the basic algorithm fails, we try all three approaches to construct good designs. Table 7 shows the methods used in the construction of good designs presented in the next section. The basic algorithm can generate all MA designs up to 40 factors for 128 runs and up to 23–28 factors for 256–4096 runs. For 128 runs, the basic algorithm can be used to construct all MA designs with

Table 8: Comparison of Some Good Designs

Run Size	Method	$n = 25$	$n = 30$	$n = 35$	$n = 40$	$n = 45$
256	Author	$A_4 = 34$	$A_4 = 93$	$A_4 = 200$	$A_4 = 370$	$A_4 = 760$
	SAS	$A_4 = 40$	$A_4 = 119$	$A_4 = 285$	$A_4 = 580$	$A_4 =$1,010
512	Author	$A_4 = 4$	$A_4 = 22$	$A_4 = 60$	$A_4 = 133$	$A_4 = 250$
	SAS	$A_4 = 4$	$A_4 = 22$	$A_4 = 63$	$A_4 = 614$	$A_4 =$1,427
1024	Author	$A_5 = 22$	$A_5 = 152$	$A_4 = 10$	$A_4 = 34$	$A_4 = 76$
	SAS	$A_5 = 55$	$A_5 = 163$	$A_4 = 180$	$A_4 = 783$	$A_4 =$1,480
2048	Author	$A_6 = 119$	$A_6 = 677$	$A_5 = 121$	$A_5 = 331$	$A_5 = 673$
	SAS	$A_6 = 139$	$A_6 = 690$	$A_5 = 112$	$A_5 = 351$	–
4096	Author	$A_6 = 15$	$A_6 = 195$	$A_6 = 856$	$A_6 =$2,086	$A_6 =$4,490
	SAS	$A_6 = 56$	$A_6 = 329$	$A_6 = 971$	$A_6 =$2,117	–

NOTE: SAS is run for up to 30 minutes on a MacBook with a 2.16 Ghz Intel Core 2 Duo CPU for each case. SAS fails to construct a resolution V design with 2048 runs for $n = 45$ and a resolution VI design with 4096 runs for $n = 45$.

some existing theories; see the next section. The first approach performs well for relatively small n while the second approach is more effective for medium to large n. In order to obtain good designs in a reasonable time, we set $M =$10,000 for resolution IV designs and $M = 1,000$ for resolution V or VI designs in the first approach and set $L =$100,000 for 256 runs and $L = 10,000$ for 512–4096 runs in the second approach. It takes about 15 and 9 minutes on a MacBook to search designs with 1024 runs and $n \leq 45$ using the first and second approach, respectively.

To determine the efficiency of our methods and designs, we use SAS PROC FACTEX to construct MA designs and compare them to ours. Table 8 lists the minimum A_4, A_5, or A_6 values for resolution IV, V, or VI designs with 256–4096 runs and n = 25, 30, 35, 40 and 45. Our designs are much better than the SAS designs except for three cases. For 512 runs and n = 25 or 30, our design has the same A_4 value as the SAS design. For 2048 runs and n = 35, our design has a larger A_5 value and is worse. In all other cases, our designs have smaller A_4, A_5 or A_6 values and thus are better. The differences are the most substantial for 1024 runs and n = 35, 40, and 45.

4 Tables of designs

Using the basic algorithm we completely enumerate all 128-run designs of resolution ≥ 4 up to 32 factors, all 256-run designs of resolution ≥ 4 up to 17 factors, all 512-run designs of resolution ≥ 5, all 1024-run designs of resolution ≥ 6, and all 2048- and 4096-run designs of resolution ≥ 7. Table 9 shows the number of nonisomorphic designs for various run sizes and resolutions. The complete set of designs can be obtained from the author upon request.

We further enumerate separately all odd 128-run designs of resolution ≥ 4, which exist for $n \leq 40$, and all even 128-run designs of resolution ≥ 4 for $n \leq 32$. For $n > 32$ all even 128-run designs of resolution IV can be obtained via their complementary even designs; see Butler (2003) and Xu and Cheng (2008). Therefore, all 128-run designs of resolution IV can be obtained. We also completely enumerate all even 256-run designs of resolution ≥ 4 for $n \leq 19$. Table 10 shows the number of nonisomorphic even and odd designs for 128, 256, 512, and 1024 runs. According to Table 10, there are 5,710 nonisomorphic even 1024-run designs of resolution ≥ 6. Draper and Mitchell (1970) identified 4,043 even designs using the letter pattern check, so they missed 1,667 (about 30%) even designs.

The 128-run designs are of special interest because MA designs are given by CSW up to 64 runs. Block and Mee (2005) constructed MA and weak MA 128-run designs for n=12–64. They achieved this by enumerating all odd designs of resolution IV and all even designs for $n \leq 22$, based on their conjecture. By comparing the numbers of even and odd designs, we conclude that their set of odd designs is complete and their set of even designs is also complete for $n \leq 22$. The numbers of even designs for $n = 21$ and 22 in their table 6 are not correct, though. So their conjecture is correct for all the cases they considered and their designs do have weak MA as claimed except for a few typos in their table 2 with 15, 19–21, and 30–32 factors (see Corrigenda). For easy reference, we give all MA and weak MA designs for 128 runs up to 40 factors in Table 11, constructed according to the procedure in Section 3.5. Note from Table 11 that MA designs are in sequential order for n=32–40. However, this is not true for $n = 31$, which agrees with the theoretical result of Xu and Cheng (2008). For $40 < n \leq 64$, MA designs can be obtained via deleting the MA complementary even designs from the unique even 2^{64-57} design; see Butler (2003), Block and Mee (2005) and Xu and Cheng (2008) for details. Again, this can be achieved by enumerating a set of good even designs. We confirm that MA designs are unique except for n=41, 42, 43, 44, and 50. For $n > 64$, MA designs can also be obtained via complementary designs; see Chen and Hedayat (1996), Tang and Wu (1996), Butler (2003), and Xu and Cheng (2008). Thus, all MA 128-run designs can be constructed.

Table 12 gives all MA and weak MA 256-run designs up to 28 factors, constructed via the basic algorithm; Table 13 lists some good designs up to 80 factors, constructed via the first two approaches described in Section 3.6. To save space, we omit a design or its generator columns in Table 13 and other tables if it can be derived from another design. For instance, designs with n=30–32 columns can be constructed as the first n columns of the design with 33 columns which are explicitly given in Table 13; designs with n=72–79 columns, not listed in Table 13, can be constructed as the first n columns of the design with 80 columns.

Block (2003) previously obtained a list of 256-run designs up to 80 factors. For $n = 24, 30, 41$–44, his designs have larger A_4 values than the designs given in Tables 12 and 13. For $n = 25$, his design is isomorphic to design 25-17.2 in Table 12 and thus does not have MA. For $n = 71$, his design has the same A_4 and A_5 values as the design given in Table 13 but has a larger A_6 value than our design. For all other cases, his designs have the same A_4, A_5 and A_6 values as the designs in Table 13. According to Xu and Cheng (2008), the designs in Table 13 have MA for $n = 69$–80. Other designs in Table 13 may not have MA.

Table 14 gives MA 512-run designs up to 25 factors and Table 15 gives some good 512-run designs up to 160 factors. These designs have resolution ≥ 6 for $n \leq 18$, resolution V for $19 \leq n \leq 23$, and resolution IV for $24 \leq n \leq 160$. The $2^{160-151}$ design is constructed by the doubling method; see Section 3.6. Draper and Mitchell (1970) conjectured that all 2^{23-14} designs of resolution V are equivalent. We confirm this; see Table 9.

Table 16 gives efficient 1024-run designs up to 45 factors. These designs have resolution ≥ 6 and MA for $n \leq 24$, resolution V for $25 \leq n \leq 33$ and resolution IV for $34 \leq n \leq 45$. The 2^{33-23} design is derived from a linear code in Chen (1991).

Table 17 gives efficient 2048-run designs up to 47 factors. These designs have resolution ≥ 7 and MA for $n \leq 23$, resolution VI for $24 \leq n \leq 34$, and resolution V for $35 \leq n \leq 47$. The 2^{34-23} design is a foldover of the 2^{33-23} design given in Table 16 and the 2^{47-36} design is derived from a linear code in Chen (1991).

Table 18 gives efficient 4096-run designs up to 65 factors. These designs have resolution ≥ 8 and MA for $n \leq 24$, resolution VI for $25 \leq n \leq 48$, and resolution V for $49 \leq n \leq 65$. The 2^{48-36} design is a foldover of the 2^{47-36} design given in Table 17 and the 2^{65-53} design is derived from a cyclic linear code in Brouwer (1998).

In Tables 11–18, each 2^{n-k} design is labeled as $n-k$ or $n-k.i$, where the index i reflects the ordering based on the MA criterion. Every 2^{n-k} design is represented by a set of n columns in the Yates order. To save space, we omit the independent columns, which are $\{1, 2, \ldots, 2^{n-k-1}\}$, and give only a set of k columns. For illustration, consider design 9-2.1 in Table 11 which has columns {31, 103}. Denote the nine factors as $\{x_1, \ldots, x_9\}$, where $\{x_1, \ldots, x_7\}$ represent independent columns, that is, $x_i = 2^{i-1}$ for $i = 1, \ldots, 7$. Then $x_8 = x_1 \cdot x_2 \cdot x_3 \cdot x_4 \cdot x_5$ and $x_9 = x_1 \cdot x_2 \cdot x_3 \cdot x_6 \cdot x_7$ because $31 = 2^0 + 2^1 + 2^2 + 2^3 + 2^4$ and $103 = 2^0 + 2^1 + 2^2 + 2^5 + 2^6$. The wordlength pattern of this design is $A_6 = 3$ and $A_i = 0$ for $i \neq 6$. If a design with n columns is not explicitly given, then it can be constructed as the first n columns of the smallest design that has more than n columns and is explicitly listed in the tables.

5 Concluding Remarks

We develop a sequential algorithm for constructing large FF designs. The new algorithm has the following features:

1. A construction procedure that allows a design to be constructed only from its MA projection in the sequential build-up process,

2. A nonisomorphism classification procedure that uses moment projection patterns to identify nonisomorphic designs efficiently,

3. A fast isomorphism check procedure that matches factors using their delete-one weight distributions,

4. A method for constructing MA designs using a partial catalog of good designs.

With some proper modifications, these features can be used to construct designs more efficiently for other situations such as blocked designs, split-plot designs, and robust parameter designs.

We further propose three approaches for constructing good designs with a large number of factors. Efficient designs are tabulated for 128–4096 runs and up to 40-160 factors. This largely extends what is available in the literature and can at least partially fulfill the increasing demand for efficient two-level FF designs with several hundred or thousand runs and dozens of factors. The construction becomes much more challenging as both the run size and the number of factors increase, which calls for further research.

Acknowledgments

The research was supported by National Science Foundation grants DMS-0505728 and DMS-0806137. The author thanks Robert Mee and Robert Block for sharing their designs and helpful communications, and Randy Sitter for sharing their manuscript. Thanks also go to the editor, an associate editor, and three referees for their constructive comments that led to an improvement of the paper.

REFERENCES

Bingham, D., and Sitter, R. R. (1999), "Minimum Aberration Two-Level Fractional Factorial Split-Plot Designs," *Technometrics*, 41, 62–70.

Block, R. M. (2003), *Theory and Construction Methods for Large Regular Resolution IV Designs*, Ph.D. dissertation, University of Tennessee, Knoxville.

Block, R. M., and Mee, R. W. (2005), "Resolution IV Designs With 128 Runs," *Journal of Quality Technology*, 37, 282–293. Corrigenda, (2006), 38, 196.

Box, G. E. P., and Hunter, J. S. (1961), "The 2^{k-p} Fractional Factorial Designs," *Technometrics*, 3, 311–351, 449–458.

Box, G. E. P., Hunter, W. G., and Hunter, J. S. (2005), *Statistics for Experimenters*, 2nd ed., New York: Wiley.

Brouwer, A. E., (1998), "Bounds on Linear Codes," *Handbook of Coding Theory*, V. S. Pless and W. C. Huffman (Eds.), New York: Elsevier, 295–461.

Butler, N. A. (2003), "Some Theory for Constructing Minimum Aberration Fractional Factorial Designs," *Biometrika*, 90, 233–238.

Chen, C. L. (1991), "Construction of Some Binary Linear Codes of Minimum Distance Five," *IEEE Transactions on Information Theory*, 37, 1429–1432.

Chen, H., and Hedayat, A. S. (1996), "2^{n-l} Designs With Weak Minimum Aberration," *The Annals of Statistics*, 24, 2536–2548.

Chen, J., and Lin, D. K. J. (1991), "On the Identity Relationship of 2^{k-p} Designs," *Journal of Statistical Planning and Inference*, 28, 95–98.

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.

Clark, J. B., and Dean, A. M. (2001), "Equivalence of Fractional Factorial Designs," *Statistica Sinica*, 11, 537–547.

Dean, A. M., and Voss, D. T. (1999), *Design and Analysis of Experiments*, New York: Springer.

Draper, N. R., and Mitchell, T. J. (1967), "The Construction of Saturated 2_{V}^{k-p} Designs," *The Annals of Mathematical Statistics*, 38, 1110–1126.

Draper, N. R., and Mitchell, T. J. (1968), "Construction of the Set of 256-Run Designs of Resolution ≥ 5 and the Set of Even 512-Run Designs of Resolution ≥ 6 With Special Reference to the Unique Saturated Designs," *The Annals of Mathematical Statistics*, 39, 246–255.

Draper, N. R., and Mitchell, T. J. (1970), "Construction of a Set of 512-Run Designs of Resolution ≥ 5 and a Set of Even 1024-Run Designs of Resolution ≥ 6," *The Annals of Mathematical Statistics*, 41, 876–887.

Franklin, M. F. (1985), "Selecting Defining Contrasts and Confounded Effects in p^{n-m} Factorial Experiments," *Technometrics*, 27, 165–172.

Franklin, M. F., and Bailey, R. A. (1977), "Selection of Defining Contrasts and Confounded Effects in Two-Level Experiments," *Applied Statistics*, 26, 321–326.

Fries, A., and Hunter, W. G. (1980), "Minimum Aberration 2^{k-p} Designs," *Technometrics*, 22, 601–608.

Hedayat, A. S., Sloane, N. J. A., and Stufken, J. (1999), *Orthogonal Arrays: Theory and Applications*, New York: Springer-Verlag.

Kleijnen, J. P. C., Sanchez, S. M., Lucas, T. W., and Cioppa, T. M. (2005), "State-of-the-Art Review: a User's Guide to the Brave New World of Designing Simulation Experiments," *INFORMS Journal on Computing*, 17, 263–289.

Lin, C. D., and Sitter, R. R. (2008), "An Isomorphism Check for Two-Level Fractional Factorial Designs," *Journal of Statistical Planning and Inference*, 138, 1085–1101.

Ma, C.-X., Fang, K.-T., and Lin, D. K. J. (2001). "On the Isomorphism of Fractional Factorial Designs," *Journal of Complexity*, 17, 86–97.

MacWilliams, F. J., and Sloane, N. J. A. (1977), *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland.

Mee, R. W. (2004), "Efficient Two-Level Designs for Estimating Main Effects and Two-Factor Interactions," *Journal of Quality Technology*, 36, 400–412.

Montgomery, D. C. (2005), *Design and Analysis of Experiments*, 6th ed., New York: Wiley.

Mukerjee, R., and Wu, C. F. J. (2006), *A Modern Theory of Factorial Designs*, New York: Springer.

Tang, B., and Wu, C. F. J. (1996), "Characterization of Minimum Aberration 2^{n-m} Designs in Terms of Their Complementary Designs," *The Annals of Statistics*, 24, 2549–2559.

Telford, J. K. (2007), "A Brief Introduction to Design of Experiments," *Johns Hopkins APL Technical Digest*, 27, 224–232.

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

Xu, H. (2003), "Minimum Moment Aberration for Nonregular Designs and Supersaturated Designs," *Statistica Sinica*, 13, 691–708.

Xu, H. (2005), "A Catalogue of Three-Level Regular Fractional Factorial Designs," *Metrika*, 62, 259–281.

Xu, H., and Cheng, C. -S. (2008), "A Complementary Design Theory for Doubling," *The Annals of Statistics*, 36, 445–457.

Xu, H., and Deng, L. -Y. (2005), "Moment Aberration Projection for Nonregular Fractional Factorial Designs," *Technometrics*, 47, 121–131.

Zhu, Y., and Zeng, P. (2005), "On the Coset Pattern Matrices and Minimum M-Aberration of 2^{n-p} Designs," *Statistica Sinica*, 15, 717–730.

Table 9: Number of Nonisomorphic Designs

	Run Size (Resolution $\geq R$)							
n	128(4)	256(4)	256(5)	512(5)	1024(6)	2048(7)	4096(7)	4096(8)
8	5							
9	13	6	5					
10	33	21	9	6				
11	92	74	11	16	6			
12	249	311	14	36	14	6		
13	623	1,429	15	92	24	9	7	6
14	1,535	7,344	11	282	47	7	17	7
15	3,522	42,581	6	1,011	98	7	27	4
16	7,500	271,784	1	4,019	185	7	48	5
17	14,438	1,798,534	1	13,759	380	3	95	5
18	25,064	?	0	29,373	919	2	113	2
19	39,335	?		31,237	1,701	1	84	1
20	57,920	?		14,135	1,682	1	35	1
21	82,496	?		2,373	739	1	22	1
22	118,444	?		128	128	1	17	1
23	173,092	?		1	8	1	17	1
24	256,654	?		0	1	0	13	1
25	376,382	?			0		0	0
26	537,907	?						
27	735,111	?						
28	956,190	?						
29	1,174,404	?						
30	1,363,003	?						
31	1,489,183	?						
32	1,535,167	?						
Total	8,948,362	?	73	96,468	5,932	46	495	35

Table 10: Number of Nonisomorphic Even and Odd Designs

	Run Size (Resolution $\geq R$)							
	128(4)		256(4)		512(5)		1024(6)	
n	Even	Odd	Even	Odd	Even	Odd	Even	Odd
8	3	2						
9	6	7	3	3				
10	14	19	9	12	3	3		
11	30	62	24	50	4	12	3	3
12	69	180	80	231	5	31	7	7
13	136	487	241	1,188	5	87	11	13
14	295	1,240	839	6,505	5	277	23	24
15	596	2,926	3,029	39,552	5	1,006	51	47
16	1,292	6,208	12,487	259,297	3	4,016	125	60
17	2,651	11,787	55,331	1,743,203	1	13,758	332	48
18	5,598	19,466	265,798	?	1	29,372	908	11
19	11,341	27,994	1,314,705	?	0	31,237	1,695	6
20	22,728	35,192	?	?		14,135	1,681	1
21	43,295	39,201	?	?		2,373	738	1
22	79,597	38,847	?	?		128	127	1
23	138,224	34,868	?	?		1	8	0
24	228,521	28,133	?	?		0	1	
25	355,813	20,569	?	?			0	
26	524,409	13,498	?	?				
27	727,036	8,075	?	?				
28	951,906	4,284	?	?				
29	1,172,255	2,149	?	?				
30	1,362,027	976	?	?				
31	1,488,750	433	?	?				
32	1,534,970	197	?	?				
33		101	?	?				
34		31	?	?				
35		13	?	?				
36		8	?	?				
37		3	?	?				
38		2	?	?				
39		1	?	?				
40		1	?	?				
Total		296,960	?	?	32	96,436	5,710	222

Table 11: Weak MA 128-Run Designs for $n \leq 40$

Design	$(A_4, A_5, \ldots)$	Columns
8-1.1	0 0 0 0 1	127
9-2.1	0 0 3 0 0	31 103
10-3.1	0 3 3 1 0	31 103 43
11-4.1	0 6 6 2 1	31 103 43 85
12-5.1	1 8 12 8	31 103 43 85 121
12-5.2	1 10 10 5	31 103 43 85 44
12-5.3	1 10 11 4	31 103 43 85 46
13-6.1	2 16 18 10	31 103 43 85 44 86
13-6.2	2 16 20 8	31 103 43 85 46 61
14-7.1	3 24 36 16	31 103 43 85 46 61 114
15-8.1	7 32 52 40	31 103 43 85 46 61 114 67
15-8.2	7 34 46 42	31 103 43 85 44 86 88 53
15-8.3	7 38 44 28	31 103 43 85 46 61 114 13
16-9.1	10 48 72	31 103 43 85 44 86 88 53 110
17-10.1	15 60 130	31 103 43 85 46 61 114 67 78 116
17-10.2	15 66 110	31 103 43 85 46 61 114 67 78 55
17-10.3	15 68 106	31 103 43 85 44 86 88 53 38 58
17-10.4	15 72 102	31 103 43 85 46 61 114 67 13 55
18-11.1	20 80 200	31 103 43 85 46 61 114 67 78 116 121
18-11.2	20 92 160	31 103 43 85 46 61 114 67 78 55 58
19-12.1	27 120 235	31 103 43 85 46 61 114 67 78 55 58 86
20-13.1	36 152 340	31 103 43 85 46 61 114 67 78 55 58 86 91
21-14.1	51 200 414	31 103 43 85 44 82 54 56 88 78 123 125 104 25
21-14.2	51 202 400	31 103 43 85 44 86 88 53 38 58 79 83 110 124
22-15.1	65 248 572	31 103 43 85 44 86 88 53 78 58 83 97 28 104 114
22-15.2	65 256 552	31 103 43 85 44 82 54 56 88 78 123 125 104 25 112
23-16.1	83 316 744	31 103 43 85 44 82 54 56 88 78 123 125 104 25 112 49
23-16.2	83 318 734	31 103 43 85 44 86 88 53 38 58 79 83 110 124 97 104
24-17.1	102 384 992	31 103 43 85 44 86 88 53 110 19 28 57 67 98 100 26 105
24-17.2	102 394 985	31 103 43 85 44 86 88 53 38 58 79 83 110 124 97 104 114
25-18.1	124 482 1312	31 103 43 85 44 86 88 53 38 58 79 83 110 124 97 104 114 123
26-19.1	152 568 1704	31 103 43 85 44 86 88 53 110 19 28 57 67 98 100 26 105 62 77
27-20.1	180 690 2200	31 103 43 85 44 86 88 53 110 19 28 57 67 98 100 26 105 62 77 112
28-21.1	210 840 2800	31 103 43 85 44 86 88 53 110 19 28 57 67 98 100 26 105 62 77 112 127
29-22.1	266 945 3472	31 103 43 85 44 86 88 53 110 19 28 57 67 98 100 26 105 62 77 112 127 124
30-23.1	335 972 4662	31 103 43 81 45 26 114 127 22 67 56 94 116 7 38 108 14 69 53 25 73 121 28
31-24.1	391 1134 5826	same as design 30-23.1, plus 91
31-24.2	391 1134 5827	same as design 30-23.1, plus 51
32-25.1	452 1322 7219	same as design 30-23.1, plus 51 97
32-25.2	452 1323 7218	same as design 30-23.1, plus 91 51
32-25.3	452 1324 7219	same as design 30-23.1, plus 51 62
33-26.1	518 1543 8863	same as design 30-23.1, plus 51 97 70
33-26.2	518 1544 8863	same as design 30-23.1, plus 91 51 62
34-27.1	589 1800	same as design 30-23.1, plus 51 97 70 79
34-27.2	589 1801	same as design 30-23.1, plus 51 97 70 87
35-28.1	665 2100	same as design 30-23.1, plus 51 97 70 79 93
35-28.2	665 2101	same as design 30-23.1, plus 51 97 70 79 91
36-29.1	756 2401	same as design 30-23.1, plus 51 97 70 79 93 62
37-30.1	854 2744	same as design 30-23.1, plus 51 97 70 79 93 62 87
38-31.1	959 3136	same as design 30-23.1, plus 51 97 70 79 93 62 87 88
39-32.1	1071 3584	same as design 30-23.1, plus 51 97 70 79 93 62 87 88 91
40-33.1	1190 4096	same as design 30-23.1, plus 51 97 70 79 93 62 87 88 91 106

Table 12: Weak MA 256-Run Designs for $n \leq 28$

Design	$(A_4, A_5, \ldots)$	Columns
9-1.1	0 0 0 0 0 1	255
10-2.1	0 0 1 2 0 0	63 199
11-3.1	0 0 6 0 1 0	127 143 179
12-4.1	0 0 12 0 3	127 143 179 213
13-5.1	0 3 12 12	127 143 179 213 105
14-6.1	0 9 18 16	127 143 179 213 105 27
15-7.1	0 15 30 26	127 143 179 213 105 27 46
16-8.1	0 24 44 40	127 143 179 85 150 75 108 189
17-9.1	0 34 68 68	127 143 179 85 150 75 108 189 229
18-10.1	3 36 114	127 143 179 213 105 27 46 182 92 194
18-10.2	3 42 102	127 143 179 213 105 27 46 152 203 214
18-10.3	3 44 98	127 143 179 213 105 27 46 92 152 203
18-10.4	3 46 92	127 143 179 85 150 75 108 189 229 7
18-10.5	3 47 95	127 143 179 213 105 27 46 77 158 185
18-10.6	3 48 94	127 143 179 213 105 27 46 77 185 234
18-10.7	3 48 96	127 143 179 85 150 75 108 118 184 234
18-10.8	3 48 102	127 143 179 213 105 27 46 158 164 185
19-11.1	4 48 168	127 143 179 213 105 27 46 182 92 194 229
19-11.2	4 56 152	127 143 179 213 105 27 46 152 203 214 236
19-11.3	4 60 144	127 143 179 213 105 27 46 152 203 214 92
19-11.4	4 64 140	127 143 179 213 105 27 46 77 158 185 234
19-11.5	4 64 152	127 143 179 213 105 27 46 158 164 185 234
20-12.1	5 64 240	127 143 179 213 105 27 46 182 92 194 229 248
20-12.2	5 80 208	127 143 179 213 105 27 46 152 203 214 236 92
21-13.1	9 104 268	127 143 179 213 105 27 46 152 203 214 236 92 45
22-14.1	14 137 346	127 143 179 213 105 27 46 77 158 185 234 164 88 201
23-15.1	20 172 450	127 143 179 213 105 27 46 77 158 185 234 201 88 43 236
24-16.1	26 216 584	127 143 179 213 105 27 46 77 158 185 84 248 166 83 146 165
25-17.1	34 262 760	127 143 179 213 105 27 46 77 158 185 84 248 166 83 146 165 49
25-17.2	34 266 752	127 143 179 213 105 27 46 77 158 185 84 248 166 83 146 165 78
26-18.1	43 325 963	127 143 179 213 105 27 46 77 158 185 84 248 166 83 146 165 78 113
26-18.2	43 326 960	127 143 179 213 105 27 46 77 158 185 84 248 166 83 146 165 78 124
27-19.1	53 395 1224	127 143 179 213 105 27 46 77 158 185 84 248 166 83 146 165 78 113 124
28-20.1	64 476 1550	127 143 179 213 105 27 46 77 158 185 84 248 166 83 146 165 78 113 124 228

Table 13: Good 256-Run Designs for $29 \leq n \leq 80$

Design	$(A_4, A_5, \ldots)$	Columns
29-21	78 579	251 244 174 99 198 27 109 92 133 87 225 73 57 137 62 51 209 148 147 44 74
30-22	93 672	
31-23	113 792	
32-24	133 932	
33-25	153 1095	239 179 188 101 149 63 75 226 121 223 216 30 25 204 182 45 142 228 53 71 118 93 115 167 176
34-26	176 1280	
35-27	200 1488	
36-28	225 1728	239 179 188 101 149 63 75 226 121 223 216 30 25 204 182 45 142 228 53 71 118 93 115 167 233 23 195 141
37-29	264 2004	
38-30	297 2304	
39-31	333 2632	
40-32	370 3008	253 230 115 186 45 89 156 240 107 62 236 55 95 153 215 146 120 57 200 222 21 134 182 212 168 207 67 101 251 138 15 161
41-33	468 3134	254 173 31 116 57 147 203 241 86 163 194 142 91 253 71 232 44 43 101 223 99 26 176 39 187 149 193 204 208 85 106 105 150
42-34	525 3516	
43-35	598 3882	247 61 91 240 30 143 106 195 86 252 79 54 210 108 180 85 101 232 201 47 177 206 138 113 56 19 174 187 250 173 41 249 151 235 126
44-36	679 4032	223 124 55 197 113 216 106 230 137 134 189 212 45 149 14 254 146 224 187 131 90 194 243 31 235 167 172 73 176 123 103 182 158 83 170 79
45-37	760 4792	
48-40	1019 6648	254 179 199 232 153 53 15 102 88 229 171 127 210 201 148 69 41 249 227 54 219 244 81 84 59 76 174 206 141 30 196 107 19 60 114 186 191 167 247 239
51-43	1365 9100	254 55 79 227 154 106 60 133 109 216 113 191 46 203 156 251 118 230 184 167 69 83 43 182 30 73 233 84 196 179 76 221 164 215 162 50 245 242 134 13 193 21 153
54-46	1769 11152	253 230 79 55 156 85 108 62 154 233 176 196 159 174 123 67 248 149 137 219 227 254 200 74 239 199 244 114 147 185 224 205 193 234 247 210 167 241 206 150 13 70 92 25 73 41
55-47	1911 12240	254 47 121 227 180 114 170 223 178 59 103 233 177 151 236 53 25 239 141 165 124 193 224 104 78 56 28 134 127 85 196 22 77 131 62 247 152 199 19 75 208 202 230 251 191 90 31
58-50	2534 15120	254 107 143 61 216 30 180 211 138 95 196 104 46 185 50 102 251 27 168 101 49 124 146 237 122 21 203 194 76 171 156 193 166 227 131 191 67 73 165 214 241 199 221 81 242 7 119 56 121 186
71-63	6273 36014	251 94 199 117 232 89 58 173 55 139 21 103 134 133 247 179 77 140 227 27 67 41 170 145 107 84 148 252 206 97 211 161 181 230 208 218 201 87 229 127 108 214 146 194 164 61 221 120 167 115 152 98 159 176 74 35 70 22 46 49 185 242 52
80-72	10300 65536	251 94 199 117 232 89 58 173 55 139 21 103 134 133 247 179 77 140 227 27 67 41 170 145 107 84 148 252 206 97 211 161 181 230 208 218 201 87 229 127 108 214 146 194 164 61 221 120 167 115 152 98 159 176 74 35 70 22 46 49 185 242 196 82 28 190 118 15 241 37 239 52

Table 14: MA 512-Run Designs for $n \leq 25$

Design	$(A_4, A_5, \ldots)$	Columns
10-1.1	0 0 0 0 0 0 1	511
11-2.1	0 0 0 2 1 0 0	127 399
12-3.1	0 0 2 4 1 0 0	127 399 179
13-4.1	0 0 4 8 3 0 0	127 399 179 341
14-5.1	0 0 7 16 7 0 0	127 399 179 341 489
15-6.1	0 0 25 0 30 0	127 391 155 301 206 501
16-7.1	0 0 44 0 45 0	127 391 155 301 206 188 358
17-8.1	0 0 68 0 85 0	127 391 155 301 206 188 358 369
18-9.1	0 0 102 0 153	127 391 155 301 206 188 358 369 468
19-10.1	0 12 84 156	127 143 307 181 211 285 327 105 427 473
20-11.1	0 16 120 240	127 143 307 181 211 285 327 105 427 473 485
21-12.1	0 21 168 360	127 143 307 181 211 285 327 105 427 473 485 510
22-13.1	0 63 189 325	127 391 155 301 206 188 358 350 507 105 298 275 369
23-14.1	0 84 252 445	127 391 155 301 206 188 358 23 340 430 435 90 450 99
24-15.1	2 102 332	127 391 155 301 206 188 358 23 469 380 226 77 441 116 420
25-16.1	4 127 428	127 143 307 181 211 285 105 427 331 337 39 456 60 453 162 198

Table 15: Good 512-Run Designs for $26 \leq n \leq 160$

Design	(A_4, A_5)	Columns
26-17	6 158	
27-18	9 195	
28-19	13 236	
29-20	17 285	501 47 440 382 242 361 422 306 189 461 401 270 427 106 343 59 480 222 487 196
30-21	22 337	
31-22	27 402	
32-23	35 470	445 118 345 199 468 59 334 156 497 356 99 430 15 85 383 153 233 165 387 443 261 339 308
33-24	43 556	445 118 345 199 468 59 334 156 497 356 99 430 15 85 383 153 233 165 387 443 261 339 172 182
34-25	52 644	493 283 174 87 107 310 220 435 201 462 340 113 416 298 269 104 406 427 181 135 31 505 469 250 496
35-26	60 756	
36-27	72 872	493 283 174 87 107 310 220 435 201 462 340 113 416 298 269 104 406 427 181 135 31 505 469 250 140 58 496
37-28	84 1004	319 482 171 327 241 109 472 510 116 23 393 282 308 353 451 313 496 154 86 391 267 429 242 471 495 334 149 75
38-29	99 1146	
39-30	115 1312	415 110 333 171 472 421 91 156 338 353 383 390 116 237 7 296 161 99 54 146 394 184 431 211 283 404 350 93 441 337
40-31	133 1484	445 115 230 295 506 31 333 403 75 297 420 245 383 414 452 488 175 368 497 101 342 92 162 306 120 206 353 168 156 125 459
41-32	153 1694	
42-33	174 1930	445 115 230 295 506 31 333 403 75 297 420 245 383 414 452 488 175 368 497 101 342 92 162 306 120 206 125 459 281 208 426 355 255
43-34	197 2184	
44-35	222 2440	439 121 227 351 496 426 220 78 201 396 114 308 255 486 364 489 119 445 338 307 56 214 178 455 19 265 348 159 39 285 417 234 262 508 408
45-36	250 2730	439 121 227 351 496 426 220 78 201 396 114 308 255 486 364 489 119 445 338 307 56 214 178 455 19 265 348 159 39 285 417 234 262 408 279 194
46-37	280 3051	
47-38	311 3411	471 122 434 302 283 191 227 462 246 177 135 43 449 77 208 492 344 357 165 496 121 110 228 408 466 397 211 323 473 388 278 31 44 50 184 332 103 506
48-39	346 3775	382 391 458 465 355 428 443 317 438 326 360 89 461 55 227 14 191 303 500 338 214 417 410 292 101 252 251 277 347 206 409 147 141 176 199 170 268 95 487
49-40	384 4175	253 286 428 471 155 313 506 372 175 105 209 204 394 71 449 478 117 243 484 339 203 354 468 328 489 438 43 432 423 319 131 268 405 302 15 133 177 151 25 230
50-41	427 4603	439 109 496 286 155 188 454 227 393 291 255 232 58 339 78 410 245 479 341 91 352 263 141 308 421 175 364 81 297 119 273 379 150 382 473 212 344 482 461 400 102

Table 15: Continued

Design	(A_4, A_5)	Columns
51-42	468 5088	501 234 121 314 364 431 220 103 479 169 480 27 166 446 504 109 85 270 158 362 419 335 215 338 428 344 127 406 284 281 410 205 441 426 46 413 293 239 219 294 139 145
52-43	521 5600	494 307 127 481 471 171 90 252 149 270 198 178 301 358 329 208 219 452 376 395 89 231 280 161 292 180 418 13 193 506 56 432 108 99 46 425 142 404 86 349 403 279 447
54-45	643 6648	477 59 391 364 410 331 229 309 207 438 86 226 119 441 452 233 279 298 69 186 134 120 491 415 291 467 139 219 500 392 74 346 345 26 431 81 37 261 369 220 286 151 148 21 416
61-52	1164 11994	239 314 457 124 406 484 419 269 240 89 503 195 476 363 86 401 236 59 302 330 324 349 475 138 504 251 90 466 61 79 226 151 167 83 444 354 454 319 157 119 229 41 133 374 359 309 396 360 52 443 510 99
68-59	1959 20034	494 181 295 223 348 118 443 470 190 449 275 363 224 140 381 237 297 166 131 202 281 290 440 378 242 358 235 115 364 323 318 209 334 196 124 43 91 511 30 249 508 501 179 44 343 438 29 101 398 386 271 325 338 353 137 278 389 266 474
78-69	3870 37963	505 370 220 279 59 397 419 195 380 267 86 490 453 190 400 420 251 178 145 30 79 503 218 169 61 294 470 467 328 184 76 346 205 213 165 392 463 357 355 217 261 88 367 126 341 74 55 270 345 457 486 386 326 333 208 480 411 305 13 385 274 21 276 142 157 405 280 151 472
87-78	6407 60906	379 399 87 481 189 358 43 90 418 203 281 439 440 309 180 50 478 365 168 117 76 155 60 460 491 297 97 292 214 262 236 284 499 349 405 304 505 488 344 410 307 319 251 84 69 465 120 279 211 46 7 322 387 19 138 484 274 267 230 310 240 508 352 339 111 370 254 455 49 37 206 335 375 106 428 30 223 457
98-89	11409 90646	379 248 341 316 94 481 141 407 239 338 454 203 54 242 396 161 329 138 181 304 224 241 418 494 477 215 314 446 374 302 45 233 191 156 186 25 369 501 468 435 506 19 97 458 227 218 292 204 295 511 400 283 125 244 351 309 108 91 365 84 484 453 323 31 436 234 150 49 385 289 112 102 26 441 478 465 421 42 79 424 463 14 122 251 135 286 352 344 11
160-151	85560 1048576	506 206 403 217 342 171 63 269 480 252 283 457 92 365 298 95 81 276 53 390 231 300 400 438 412 478 366 505 475 417 45 466 183 30 380 495 339 279 152 149 359 392 307 137 483 73 440 243 131 166 138 57 317 200 74 168 108 273 58 356 234 120 503 150 193 274 321 485 477 294 455 409 452 116 341 318 410 212 465 395 240 98 486 155 255 140 97 46 420 346 7 389 178 500 430 180 228 29 190 415 246 293 165 270 123 336 437 177 345 126 215 322 423 332 418 429 69 189 218 335 472 303 370 11 70 458 125 19 54 369 297 304 360 82 119 60 492 35 205 383 67 280 363 111 233 203 443 194 143 259 245

Table 16: Efficient 1024-Run Designs for $n \leq 45$

Design	$(A_4, A_5, \ldots)$	Columns
11-1.1	0 0 0 0 0 0	1023
12-2.1	0 0 0 0 3 0	127 911
13-3.1	0 0 0 4 3 0	127 911 435
14-4.1	0 0 0 8 7 0	127 911 435 725
15-5.1	0 0 0 15 15	127 911 435 725 873
16-6.1	0 0 6 25 15	127 911 435 725 873 158
17-7.1	0 0 12 41	127 911 435 725 873 158 327
18-8.1	0 0 19 66	127 911 435 725 873 158 327 490
19-9.1	0 0 28 104	127 911 435 725 873 158 327 490 626
20-10.1	0 0 40 160	127 911 435 725 873 158 327 490 626 697
21-11.1	0 0 56 240	127 911 435 725 873 158 327 490 626 697 860
22-12.1	0 0 77 352	127 911 435 725 873 158 327 490 626 697 860 932
23-13.1	0 0 251 0	127 911 179 341 614 158 968 283 805 466 555 508 535
24-14.1	0 0 336 0	127 911 179 341 614 158 790 440 964 625 995 234 334 589
25-15	0 22 336	
26-16	0 44 358	439 892 174 213 847 109 647 299 985 31 805 716 339 701 222 231
27-17	0 68 392	415 892 174 631 901 107 94 493 295 716 314 527 793 697 940 605 586
28-18	0 90 483	415 892 174 631 901 107 94 493 295 716 314 527 793 753 676 455 270 669
29-19	0 118 586	638 1009 283 982 244 844 541 174 809 121 677 835 453 474 615 207 960 503 92
30-20	0 152 703	998 127 433 861 527 714 681 286 299 908 376 220 242 135 975 763 189 610 836 938
31-21	0 189 863	638 1009 283 982 244 844 541 174 809 121 677 835 453 670 54 401 169 621 778 390 957
32-22	0 231 1056	
33-23	0 275 1287	979 877 351 1012 391 362 213 974 307 905 412 811 639 114 187 688 202 92 669 572 848 870 534
35-25	10 365	
38-28	22 564	702 493 865 422 722 283 1002 405 251 953 47 1015 899 882 101 163 474 736 554 563 733 879 57 383 151 651 71 646
40-30	34 728	
42-32	48 940	859 757 227 557 914 124 988 463 214 678 488 939 154 46 423 1023 867 876 396 630 537 583 696 805 921 712 822 169 500 754 618 743
45-35	76 1344	830 967 428 475 220 182 601 844 654 703 519 298 636 353 29 969 929 565 918 875 111 725 755 624 307 456 399 832 792 805 386 506 366 62 1008

Table 17: Efficient 2048-Run Designs for $n \leq 47$

Design	$(A_5, A_6, \ldots)$	Columns
12-1.1	0 0 0 0 0 0	2047
13-2.1	0 0 0 1 2 0	255 1807
14-3.1	0 0 0 7 0 0	127 911 1459
15-4.1	0 0 0 15 0 0	127 911 1459 1749
16-5.1	0 0 0 30 0 0	127 911 1459 1749 1897
17-6.1	0 0 16 30 0	127 911 1459 1749 1897 470
18-7.1	0 0 32 46 0	127 911 1459 1749 1897 470 739
19-8.1	0 0 52 78 0	127 911 1459 1749 1897 470 739 826
20-9.1	0 0 80 130	127 911 1459 1749 1897 470 739 826 1272
21-10.1	0 0 120 210	127 911 1459 1749 1897 470 739 826 1272 1309
22-11.1	0 0 176 330	127 911 1459 1749 1897 470 739 826 1272 1309 1614
23-12.1	0 0 253 506	127 911 1459 1749 1897 470 739 826 1272 1309 1614 1956
24-13	0 85 272	
25-14	0 119 336	
26-15	0 166 416	
27-16	0 230 512	1646 1438 247 1748 1227 635 1807 494 1356 969 1964 854 2047 1299 1521 1378
28-17	0 421 0	1946 1863 1001 1660 757 671 433 1361 626 254 1221 551 2004 1078 1549 397 1402
29-18	0 537 0	1598 743 1931 1367 426 1473 696 1144 358 1754 1749 635 1797 1615 1858 1012 1500 179
30-19	0 677 0	1646 823 1494 1693 109 682 968 1869 1243 1294 1330 1009 494 628 991 1872 376 1067 1453
31-20	0 845 0	1722 379 935 1623 494 1426 223 527 1294 962 1670 637 1816 248 1081 1483 1500 1453 409 747
32-21	0 1048 0	
33-22	0 1285 0	
34-23	0 1562 0	743 954 1387 1524 1077 880 1417 2047 421 1001 214 790 1454 618 1082 1574 1709 1736 313 1474 731 654 913
35-24	121 1069	
40-29	331 2170	1822 1645 1012 1423 1187 555 1180 1137 2027 453 1990 151 1528 295 818 362 185 1321 1356 94 1607 1713 825 1394 1297 804 848 1198 243
45-34	673 4493	
47-36	846 5922	1243 995 381 1654 1993 1063 916 1888 1550 2002 1624 719 823 572 1180 689 1299 890 1342 183 1635 726 269 350 337 1254 1532 1080 1998 1352 1237 1978 1443 531 1701 306

Table 18: Efficient 4096-Run Designs for $n \leq 65$

Design	$(A_5, A_6, \ldots)$	Columns
13-1.1	0 0 0 0 0 0	4095
14-2.1	0 0 0 0 2 1	511 3615
15-3.1	0 0 0 3 4 0	255 1807 2867
16-4.1	0 0 0 7 8 0	255 1807 2867 3413
17-5.1	0 0 0 14 16	255 1807 2867 3413 3734
18-6.1	0 0 0 45 0	2047 2111 2503 2777 2922 3308
19-7.1	0 0 0 78 0	2047 2111 2503 2777 2922 3308 2996
20-8.1	0 0 0 130 0	2047 2111 2503 2777 2922 3308 2996 3441
21-9.1	0 0 0 210 0	2047 2111 2503 2777 2922 3308 2996 3441 3482
22-10.1	0 0 0 330 0	2047 2111 2503 2777 2922 3308 2996 3441 3482 3670
23-11.1	0 0 0 506 0	2047 2111 2503 2777 2922 3308 2996 3441 3482 3670 3747
24-12.1	0 0 0 759 0	2047 2111 2503 2777 2922 3308 2996 3441 3482 3670 3747 3853
25-13	0 15 196	4031 3914 1657 2771 1507 2552 746 347 2913 3273 3186 2206 2693
26-14	0 36 249	
27-15	0 57 309	4031 3914 1657 2771 1507 2552 2841 2978 1675 1338 3249 2964 238 2125 3212
28-16	0 90 396	4031 3429 471 889 3922 3289 303 3622 3459 3384 946 2385 2540 1721 2884 2700
29-17	0 130 488	4031 3914 1657 2771 1507 2917 2552 2841 2978 1675 1338 350 742 3158 3533 1984 301
30-18	0 195 544	4031 3914 1657 2771 1507 750 2552 3190 2841 1675 3249 3565 4048 3334 3649 2204 1428 3644
31-19	0 282 633	4088 1231 3254 3427 4038 1883 1457 1587 3245 2961 3353 3700 2764 3819 3610 3404 957 2142 996
32-20	0 402 448	3821 1522 1931 949 223 984 3726 2899 2077 2486 2706 3834 1918 2859 3205 3985 484 3611 3196 2360
35-23	0 856 0	3511 890 2734 253 1623 3988 3166 1349 1955 2465 1657 913 671 2789 2119 973 3683 214 1981 2927 3652 1772 3964
40-28	0 2086 0	
41-29	0 2460 0	3447 719 1438 2965 4076 1417 2874 2246 3373 823 2004 2497 744 2478 3506 2698 3647 701 3277 1082 1479 3172 3815 3947 997 3091 2356 2911 1952
45-33	0 4490 0	
48-36	0 6768 0	4087 2862 1468 2235 3530 1658 866 3451 1707 3166 3589 1423 2601 1684 551 3656 3851 3556 2557 3968 3254 1735 3345 604 1893 3048 471 2630 2869 3313 491 1585 2840 2435 3803 2249
65-53	2223 21840	3006 1245 3924 2159 1441 2808 815 1721 3352 1887 1653 1234 3560 3659 1063 3590 2545 457 997 915 3300 3823 1731 2508 1098 3665 860 3132 1458 429 3382 1555 609 1433 3087 2227 2435 3747 3776 2313 2423 1531 4083 3990 1931 2402 3332 2828 1581 1910 1758 219 1018