August 2013

Constructing Orthogonal Arrays on Non-abelian Groups

Margaret Ann McComack
University of Wisconsin-Milwaukee

Constructing Orthogonal Arrays on Non-abelian Groups

by

Margaret Ann McComack

A Thesis Submitted in
Partial Fulfillment of the
Requirements for the Degree of

Master of Science
in Mathematics

at

The University of Wisconsin-Milwaukee
August 2013

ABSTRACT

Constructing Orthogonal Arrays on Non-abelian Groups

by

Margaret Ann McComack

The University of Wisconsin-Milwaukee, 2013
Under the Supervision of Professor Jay H. Beder

For an orthogonal array (or fractional factorial design) on k factors, Xu and Wu (2001) define the array's *generalized wordlength pattern*, $(A_1, \ldots, A_k)$, by relating a cyclic group to each factor. They prove the property that the array has strength t if and only if $A_1 = \cdots = A_t = 0$. In their 2012 paper, Beder and Beder show that this result is independent of the group structure used. Non-abelian groups can be used if the assumption is made that the groups G_i are chosen so that the counting function O of the array is a class function on G. The aim of this thesis is to construct examples of orthogonal arrays on $G = G_1 \times \cdots \times G_k$, where G is non-abelian, having two properties: given strength, and counting function O that is constant on the conjugacy classes of G.

Table of Contents

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Jay H. Beder, for his guidance on this thesis. Dr. Beder has been extremely accommodating in all aspects of the thesis. He has been flexible with meetings, understanding of personal conflicts, helpful in teaching new concepts and programs, and patient throughout all of it. It has been a joy to work with Dr. Beder. I could not have dreamed of a better advisor.

In addition I would like to thank Dr. Jeb Willenbring for his instruction and programming of Maple. Without his help, confirming the condition of constancy on conjugacy classes would have been incredibly difficult. I would like to thank Dr. Yi Ming Zou and Dr. Richard Stockbridge for being part of the committee and for giving their time towards this thesis.

Thank you to all my past professors who helped me in my education, particularly Dr. Thomas Weber. Without his help, guidance, and kindness I would not be where I am today.

Lastly, I would like to thank my family and friends for their constant support. Thank you to my parents David and Catherine, and my sister Joanna, for always believing in me and encouraging me through this whole process. Thank you to my wonderful husband Joshua for his love, support, and patience as I completed my degree. He gave me the motivation and the drive to finish so I can start a new chapter in my life.

Chapter 1

Introduction

An **orthogonal array**, or **design**, is a subset D of a Cartesian product $G = G_1 \times \cdots \times G_k$ with possible repetitions – a multisubset. It has a particular parameter, its **strength**, which we define in Chapter 2. We will often refer to an orthogonal array as simply an **array**.

Xu and Wu [2] define the *generalized wordlength pattern*, $(A_1, \ldots, A_k)$, of an array by indexing the levels of each factor by a cyclic group, so that G is itself a group. They prove that the array has strength t if and only if $A_1 = \cdots = A_t = 0$. Beder and Beder [1] determine that this result does not depend on the group structure. Non-abelian groups can be used if an assumption is made about the counting function O of the array, namely that the counting function is constant on the conjugacy classes of G. We will refer to these arrays as **conjugacy arrays**. Two examples are given in [1], which use the non-abelian group S_3. The purpose of this thesis is to construct more examples using at least one G_i that is non-abelian.

For the purpose of this thesis, factors of the groups considered for the construction of an array are of order less than or equal to 10. The groups have 3 to 6 factors, the majority having 3 or 4, and all arrays are constructed to have strength $k - 1$. Minitab as well as a program created in Maple were used to assist in checking the two properties of the array, its strength and the constancy of O on conjugacy classes. The procedure for the Minitab program and the Maple code are found in Appendix A and B respectively.

Chapter 2 defines an orthogonal array, along with associated terms. The groups used in constructing the arrays and the conjugacy classes of the groups used are the focus of Chapter 3. Chapter 4 and Chapter 5 describe the process of constructing arrays in the general case, as well as giving actual examples.

Notation to be used in this thesis:

- $\mathbb{Z}_n$ for the integers mod n. We write $\mathbb{Z}_n = \{0, \ldots, n-1\}$.
- S_n for the symmetric group of degree n. This is the group of permutations on n symbols, which we will take to be $1, \ldots, n$.
- Dih_n for the dihedral group of order $2n$
- Q_8 for the quaternion group of order 8
- $|E|$ for the cardinality of the set E
- k for the number of factors of a cartesian product
- t for the strength of an array
- lcm for the least common multiple
- gcd for the greatest common divisor
- As usual, $a|b$ ("a divides b") means that a is a factor of b

Chapter 2

Orthogonal Arrays

2.1 Definitions

A **factorial experiment** (or full factorial design) is an experiment with k factors such that each factor has a finite set of levels. For the set $G = G_1 \times \cdots \times G_k$ where the set G_i is finite and indexes the order of the ith factor, let

$$s_i = |G_i|. \tag{2.1}$$

Thus there are $s_1 \times \cdots \times s_k$ treatment combinations given by the k-tuples in the set. We will call s_i the **order** of the ith factor.

An **orthogonal array** (or **fractional factorial design**) is a multisubset, a subset in which elements can be repeated, of the set G. Let the multisubset be called D, and let $O(x)$ be the number of times the k-tuple x appears in D. O is then the **counting function** of D. The **size** of the array is then

$$N = |D| = \sum_{x \in G} O(x), \tag{2.2}$$

Using this, an orthogonal array can be presented as a $k \times N$ matrix, where the columns represent the elements in the array. The goal of this thesis is to find examples of orthogonal arrays indexed by non-abelian groups, where the array has two properties: given strength, and a counting function that is constant on conjugacy classes. From this point forward we will refer to this type of array as a **conjugacy array.**

Let the array D have factors 1, …, k, represented by rows. Choosing m rows, $i_1, \ldots, i_m$, results in another orthogonal array, say D'. The array D' is a multisubset of $G' = G_{i_1} \times \cdots \times G_{i_m}$, and has its own counting function O'. We call D' the **projection** of D on factors $i_1, \ldots, i_m$.

The array D has **strength** t if the projection D' of D on any t factors, $i_1, \ldots, i_t$, consists of λ_I copies of $G_{i_1} \times \cdots \times G_{i_t}$ for some integer λ_I where $I = \{i_1, \ldots, i_t\}$. If an array has strength t then it has strength p for all $p \leq t$. In particular, in an array of strength 1 the ith row has λ_i copies of G_i for some integer λ_i.

Consider an array with strength t. For each $I \subset \{1, \ldots, k\}$ with $|I| = t$, the projection D'_I has a constant counting function, O'_I. That is $O'_I(y) \equiv \lambda_I$ for all $y \in \prod_{i \in I} G_i$. The size of the array can now be represented as

$$N = \lambda_I \prod_{i \in I} s_i. \tag{2.3}$$

Therefore N needs to be a multiple of all possible products of t orders and therefore a multiple of

$$L_t = \operatorname{lcm} \left\{ \prod_{i \in I} s_i : |I| = t \right\}. \tag{2.4}$$

This makes L_t the minimum size of the array.

An orthogonal array is **symmetric** if $s_1 = \cdots = s_k$. The common order is denoted as s. If a symmetric array D has strength t, then there is a common λ which is called the *index* of the array. In this paper we will mainly consider asymmetric arrays.

The set $G = G_1 \times \cdots \times G_k$ itself is an orthogonal array with strength k. This is a trivial example. The purpose of this thesis is to construct orthogonal arrays that are **proper fractions** of G, ones that are not one or more copies of G itself. From this point forward, when an array is a copy of G itself, we will refer to it as the **complete design** or sometimes as **an** $s_1 \times \cdots \times s_k$ **design**. A fraction that is, say, half of the complete design will be called a $\frac{1}{2}$ **fraction**. If the fraction is a conjugacy array we will call it a $\frac{1}{2}$ **conjugacy array**.

2.2 Trivial Cases

When constructing the examples for this thesis, there is a condition on the order of the factors that leads to the smallest possible array being the size of the complete design. We will call this a **trivial case**. Such an array has size L_k since L_k is the product $s_1 \cdots s_k$. Proposition 2.2.1 below shows a condition on the levels s_i that will result in the trivial case. Since we only consider arrays of strength $k-1$, we also give a sufficient condition for $L_{k-1} < L_k$.

Lemma 1.

For any array, $L_1 \leq \ldots \leq L_k$, and in fact $L_i | L_j$ for $i < j$.

Proof. Suppose p is a prime and $p^k | L_t$ then $p^k | \prod_{i \in I} s_i$ for some I such that $|I| = t$. Then $p^k | \prod_{i \in J} s_i$ for some $J \supset I$, where $|J| = t+1$. Therefore $p^k | L_{t+1}$.

Since $p^k | L_t$ implies $p^k | L_{t+1}$ for all primes p, it follows that $L_t | L_{t+1}$. □

Lemma 1 lead to a condition on the levels s_i that determines for what values of t, L_t is the size of the complete design.

Proposition 2.2.1.

For levels $s_1, \ldots, s_k$, let $e_I = \gcd\{s_i, i \in I\}$ for $I \subseteq \{1, \ldots, k\}$. Let $d = \max\{|I| : e_I > 1\}$. Then $L_t = L_k$ if $t \geq d$. If $d = k$ then $L_{k-1} < L_k$.

Proof. Let $s_1, \ldots, s_d$ be levels with common factor e_I. According to Lemma 1 it suffices to show that $L_t = L_{t+1}$ for $t \geq d$.

To prove this we fix a set J with $|J| = t+1$ and consider all $I \subset J$ with $|I| = t$. Using the fact that for integers $a_1, \ldots, a_n$,

$$\operatorname{lcm}\left(\prod_{i \in I} a_i, |I| = n-1\right) = \prod_{i=1}^{n} a_i / \gcd(a_1, \ldots, a_n),$$

we can write

$$\operatorname{lcm}\left(\prod_{i \in I} s_i, |I| = t, I \subset J\right) = \prod_{i \in J} s_i / \gcd\{s_i, i \in J\}. \tag{2.5}$$

Since $|J| > d$ it follows that $\gcd\{s_i, i \in J\} = 1$. Therefore we can write

$$\operatorname{lcm}\left(\prod_{i\in I} s_i, |I| = t, I \subset J\right) = \prod_{i\in J} s_i.$$

For each J with $|J| = t + 1$, consider

$$A_J = \left\{\prod_{i\in I} s_i, |I| = t, I \subset J\right\}.$$

Then

$$A = \bigcup_J A_J = \left\{\prod_{i\in I} s_i, |I| = t\right\}.$$

We know that

$$L_t = \operatorname{lcm}\left(\prod_{i\in I} s_i, |I| = t\right) = \operatorname{lcm}(A),$$

and using the fact that, if A is a set of integers and $A = A_1 \cup \cdots \cup A_r$, then

$$\operatorname{lcm}(A) = \operatorname{lcm}(\operatorname{lcm}(A_1), \ldots, \operatorname{lcm}(A_r)),$$

we have

$$\begin{aligned} L_t &= \operatorname{lcm}\left(\operatorname{lcm}(A_J), \text{all } |J| = t+1\right) \\ &= \operatorname{lcm}\left(\prod_{i\in J} s_i, |J| = t+1\right) \\ &= L_{t+1}. \end{aligned}$$

We have shown that $L_t = L_{t+1}$ for $t \geq d$.

Now assume $d = k$. We consider all $I \subset J$ with $J = \{1, \ldots, k\}$ and $|I| = k - 1$. Using (2.5) we can write

$$L_{k-1} = \operatorname{lcm}\left(\prod_{i\in I} s_i, |I| = k-1, I \subset J\right) = \prod_{i\in J} s_i / \gcd\{s_i, i = 1, \ldots, k\}.$$

We know

$$L_k = \prod_{i\in J} s_i$$

Since $d = k$ it follows that $\gcd(s_1, \ldots, s_k) > 1$. Then we see that

$$\prod_{i \in J} s_i / \gcd\{s_i, i \in J\} < \prod_{i \in J} s_i.$$

We have shown $L_{k-1} < L_k$ when $d = k$. □

When $d = k$ the levels $s_1, \ldots, s_k$ have a common factor. This is true for all examples constructed for this thesis. Proposition 2.2.1 determines the largest value of t that can result in an array that is not the size of the complete design.

When constructing an array of strength t, the smallest constructible array is not always of minimum size L_t. Steps for constructing the arrays of minimum size will be discussed in detail in Chapter 4.

Chapter 3

Groups Used in Construction

If $G_1, \ldots, G_k$ are groups, the direct product $G = G_1 \times \cdots \times G_k$ is non-abelian if and only if at least one G_i is non-abelian. In this thesis, unless otherwise stated, all examples are constructed using only one non-abelian group. Without loss of generality, the non-abelian group will be G_1. The numbers s_i will be arranged in descending order. Therefore the factor with the highest order will be the non-abelian factor.

3.1 Conjugacy

Let g and h be elements of a group G. The elements g and h are **conjugate** if $h = xgx^{-1}$ for some $x \in G$. Conjugacy is an equivalence relation on G, and thus the equivalence classes are a partition of G. These classes are called **conjugacy classes**.

Lemma 2.

If $G = G_1 \times \cdots \times G_k$ then the conjugacy classes of G are of the form $C_1 \times \cdots \times C_k$ where C_i is a conjugacy class of G_i.

Proof. Denoting conjugacy in G by $\sim$ and conjugacy in G_i by $\sim_i$ it is enough to show:

$$(x_1, \ldots, x_k) \sim (y_1, \ldots, y_k) \text{ iff } x_i \sim_i y_i,\ i = 1, \ldots, k.$$

But

$$\begin{aligned} x_i \sim_i y_i \ \forall i \Leftrightarrow \forall i \ \exists z_i \in G_i \text{ such that } x_i &= z_i y_i {z_i}^{-1} \\ \Leftrightarrow (x_1, \dots, x_k) &= (z_1 y_1 {z_1}^{-1}, \dots, z_k y_k {z_k}^{-1}) \\ &= (z_1 \dots z_k)(y_1 \dots y_k)({z_1}^{-1} \dots {z_k}^{-1}) \\ &= (z_1 \dots z_k)(y_1 \dots y_k)(z_1 \dots z_k)^{-1} \\ \Leftrightarrow (x_1, \dots, x_k) &\sim (y_1, \dots, y_k) \end{aligned}$$

□

To list the conjugacy classes of G, we find a representative g_i of C_i for each i. Then $(g_1, \dots, g_k)$ represents $C_1 \times \dots \times C_k$.

In an abelian group each element is its own conjugacy class. Therefore all the elements are singleton conjugacy classes. This is convenient for constructing orthogonal arrays with the given properties.

3.2 Abelian Groups

In the construction of orthogonal arrays, abelian and non-abelian groups will be used. The abelian groups range from order two to order six, and are the additive cyclic groups $\mathbb{Z}_n$.

The group $\mathbb{Z}_5$ is used in just one example, where the method of construction leads to a trivial case. This example is shown in Chapter 5.

3.3 S_3: Symmetric Group of Order Six

The first non-abelian group used in constructing orthogonal arrays is S_3, because it is the smallest.

We let $S_3 = \{e, x, y, a, b, c\}$ with permutations shown below:

- $e =$ the identity
- $x = (1\ 2\ 3)$

- $y = (1\ 3\ 2)$
- $a = (2\ 3)$
- $b = (1\ 3)$
- $c = (1\ 2)$

The permutations are represented by letters for easier representation in the array. S_3 is the group of symmetries of an equilateral triangle, where e is the identity element, x and y are the rotations, and a, b, and c are the reflections. It has three conjugacy classes:

$$\{e\}, \{x, y\}, \text{and}\{a, b, c\}. \tag{3.1}$$

3.4 Dih_4: Dihedral Group of Order Eight

There are two non-abelian groups of order eight, Dih_4, the dihedral group of order eight, and Q_8, the quaternion group.

We let $Dih_4 = \{e, q, r, s, a, b, x, y\}$ with permutations shown below:

- $e =$ the identity
- $q = (1\ 3)(2\ 4)$
- $r = (1\ 2\ 3\ 4)$
- $s = (1\ 4\ 3\ 2)$
- $a = (1\ 3)$
- $b = (2\ 4)$
- $x = (1\ 4)(2\ 3)$
- $y = (1\ 2)(3\ 4)$

Dih_4 is the group of symmetries of a square and has eight elements. In relation to the square, e is the identity element, q is the half turn rotation, r and s are the quarter and three quarter turn rotations respectively, a and b are reflections about the diagonals, and x and y are the reflections about the lines joining the midpoints of opposite sides. The non-abelian group Dih_4 has five conjugacy classes:

$$\{e\}, \{q\}, \{r, s\}, \{a, b\}, \text{and}\{x, y\}. \tag{3.2}$$

The non-abelian group Q_8 has five conjugacy classes. There are three classes with two elements and two classes with one element. Since Q_8 and Dih_4 have the same number of conjugacy classes and the class sizes are the same, using Q_8 to construct examples of orthogonal arrays is unnecessary: replacing the elements of Dih_4 with the elements of Q_8 will result in identically structured arrays.

3.5 Dih_5: Dihedral Group of Order Ten

The last non-abelian group considered is Dih_5, being the third smallest non-abelian group. The dihedral group of order ten is the group of symmetries of a pentagon, and is a subgroup of S_5.

We let $Dih_5 = \{e, a, b, c, d, q, w, x, y, z\}$ with permutations show below:

- $e =$ the identity
- $a = (1\ 2\ 3\ 4\ 5)$
- $b = (1\ 3\ 5\ 2\ 4)$
- $c = (1\ 4\ 2\ 5\ 3)$
- $d = (1\ 5\ 4\ 3\ 2)$
- $q = (2\ 5)(3\ 4)$
- $w = (1\ 3)(4\ 5)$
- $x = (1\ 5)(2\ 4)$

- $y = (1\ 2)(3\ 5)$
- $z = (1\ 4)(2\ 3)$

In relation to the symmetries of the pentagon, e is the identity element, a, b, c, and d are the rotations, and q, w, x, y, and z are the reflections about a line from a vertex to the midpoint of the opposite side.

The group Dih_5 has four conjugacy classes

$$\{e\}, \{a, d\}, \{b, c\}, \text{and}\{q, w, x, y, z\}, \tag{3.3}$$

where $\{a, d\}$ are the rotations of $\pm 72\,^\circ$, and $\{b, c\}$ are the rotations of $\pm 144\,^\circ$.

Chapter 4

Arrays of Minimum Size

4.1 Construction Techniques

In this chapter we will describe the process of constructing a conjugacy array of strength $k-1$ and minimum size L_{k-1} on $G = G_1 \times \cdots \times G_k$. Each step will be described in the general sense, and then a specific example will be provided. As before, $s_i = |G_i|$. In all cases in this chapter the orders s_i have a common factor of 2 or 4.

We construct the array as a $k \times N$ matrix. Row i will contain the elements of G_i, repeated in certain patterns. We will adopt a fixed order for the elements of the groups used in construction. The fixed order for each group is shown below.

- $S_3 - e, x, y, a, b, c$
- $Dih_4 - e, q, r, s, a, b, x, y$
- $Dih_5 - e, a, d, b, c, q, w, x, y, z$
- $\mathbb{Z}_n - 0, 1, 2, \ldots, n-1$

Note that conjugate elements are adjacent in these orderings, as can be seen in (3.1), (3.2) and (3.3).

Since L_{k-1} is a multiple of all products of s_i of size $k-1$, each s_i must divide it.

It is natural for us to begin by seeking arrays of size L_{k-1}. We define

$$\begin{aligned} v_1 &= \frac{L_{k-1}}{s_1} \\ v_2 &= \frac{L_{k-1}}{s_2} \\ v_j &= \frac{L_{k-1}}{s_2 \cdots s_j} \text{ for } j > 2. \end{aligned}$$

We note also that in the constructions using S_3 and the dihedral groups, the first v_k elements of G_1 will form a union of conjugacy classes, and that the array constructed is a v_k/s_1 fraction.

4.2 Construction of a $\frac{1}{2}$ Conjugacy Array

In this section we assume that the orders s_i have common factor 2. The conjugacy array is $\frac{1}{2}$ the size of the complete design.

To construct the first row of the matrix we write the elements of G_1 in the fixed order. We repeat that arrangement v_1 times to create the first row of the array.

To construct the second row, write the first element of G_2 v_2 times. This is repeated with the remaining elements of G_2, keeping the elements in the fixed order.

For rows 3 through $(k-1)$, row j has each element of G_j repeated v_j times, keeping the elements in the fixed order. This pattern of elements is repeated as much as necessary to fill the row.

To construct the kth row, each element of G_k is repeated v_k times, the whole pattern repeated as often as necessary to fill the row. This is subject to a special condition: the elements are listed in the fixed order in the first segment, and in reverse fixed order in the second segment. The segments alternate thereafter to fill the row.

Example 1. To illustrate a $\frac{1}{2}$ fraction with 3 factors we use the group $G = S_3 \times \mathbb{Z}_4 \times \mathbb{Z}_2$:

We have $L_2 = 24$. Then $v_1 = \frac{24}{6} = 4$

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c \end{pmatrix}$$

$v_2 = \frac{24}{4} = 6$

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 3 & 3 & 3 \end{pmatrix}$$

We have $v_k = v_3 = \frac{L_2}{s_2 s_3} = \frac{24}{8} = 3$. The array has 4 segments of length 6. For the first segment, rows 2 and 3 are:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

For the second segment they are:

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

For this example we have the $\frac{1}{2}$ fraction:

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 3 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Example 2. To illustrate a $\frac{1}{2}$ fraction with 4 factors we use the group $G = S_3 \times \mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_2$ and show the construction below. This example uses the same group and is the same size as an example constructed in [1], but is constructed in the method described in Section 4.2, and results in a different fraction of the complete design.

We have $L_3 = 24$. Then $v_1 = \frac{24}{6} = 4$

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c \end{pmatrix}$$

$v_2 = \frac{24}{2} = 12$

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$v_3 = \frac{24}{4} = 6$

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c \\ 0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&1&1&1&1 \\ 0&0&0&0&0&0&1&1&1&1&1&1&0&0&0&0&0&0&1&1&1&1&1&1 \end{pmatrix}$$

We have $v_k = v_4 = \frac{L_3}{s_2 s_3 s_4} = \frac{24}{8} = 3$. The array has 4 segments of length 6. For the first segment, rows 3 and 4 are:

$$\begin{pmatrix} 0&0&0&0&0&0 \\ 0&0&0&1&1&1 \end{pmatrix}$$

For the second segment they are:

$$\begin{pmatrix} 1&1&1&1&1&1 \\ 1&1&1&0&0&0 \end{pmatrix}$$

For this example we have the $\frac{1}{2}$ fraction:

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c \\ 0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&1&1&1&1 \\ 0&0&0&0&0&0&1&1&1&1&1&1&0&0&0&0&0&0&1&1&1&1&1&1 \\ 0&0&0&1&1&1&1&1&1&0&0&0&0&0&0&1&1&1&1&1&1&0&0&0 \end{pmatrix}$$

Below are a few other $\frac{1}{2}$ fractions constructed for this thesis. The first example uses the same group and is the same size as an example constructed in [1], but is constructed in the method described in Section 4.2. This also results in a different fraction of the complete design than the one constructed in [1].

$G = S_3 \times \mathbb{Z}_2 \times \mathbb{Z}_2$

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c \\ 0&0&0&0&0&0&1&1&1&1&1&1 \\ 0&0&0&1&1&1&1&1&1&0&0&0 \end{pmatrix} \tag{4.1}$$

$G = Dih_4 \times \mathbb{Z}_2 \times \mathbb{Z}_2$

$$\begin{pmatrix} e & q & r & s & a & b & x & y & e & q & r & s & a & b & x & y \\ 0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1 \\ 0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&0 \end{pmatrix}$$

$G = Dih_5 \times \mathbb{Z}_2 \times \mathbb{Z}_2$

$$\begin{pmatrix} e & a & b & c & d & q & w & x & y & z & e & a & b & c & d & q & w & x & y & z \\ 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&1&1 \\ 0&0&0&0&0&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0 \end{pmatrix}$$

4.3 Construction of a $\frac{1}{4}$ Conjugacy Array

In this section we assume that factors s_i have common factor 4. The conjugacy array is $\frac{1}{4}$ the size of the complete design. The construction method is identical to that shown in Section 4.2.

There are two examples constructed for this thesis that yield $\frac{1}{4}$ fractions: $G = Dih_4 \times \mathbb{Z}_4 \times \mathbb{Z}_4$ and $G = Dih_4 \times \mathbb{Z}_4 \times \mathbb{Z}_4 \times \mathbb{Z}_4$.

To illustrate a $\frac{1}{4}$ fraction with 3 factors we use the group $G = Dih_4 \times \mathbb{Z}_4 \times \mathbb{Z}_4$:

- We have $L_2 = 32$. Then $v_1 = \frac{32}{8} = 4$.
- $v_2 = \frac{32}{4} = 8$
- We have $v_k = v_3 = \frac{L_2}{s_2 s_3} = \frac{32}{16} = 2$. The array has 4 segments of length 8. For the first segment, rows 2 and 3 are:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 2 & 2 & 3 & 3 \end{pmatrix}$$

  For the second segment they are:

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 3 & 3 & 2 & 2 & 1 & 1 & 0 & 0 \end{pmatrix}$$

- The array is the juxtaposition of four segments.

$$\begin{pmatrix} e & q & r & s & a & b & x & y & e & q & r & s & a & b & x & y & e & q & r & s & a & b & x & y & e & q & r & s & a & b & x & y \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 \\ 0 & 0 & 1 & 1 & 2 & 2 & 3 & 3 & 3 & 3 & 2 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 2 & 2 & 3 & 3 & 3 & 3 & 2 & 2 & 1 & 1 & 0 & 0 \end{pmatrix}$$

4.4 Increasing Strength

It is possible to take any $\frac{1}{2}$ or $\frac{1}{4}$ fraction constructed in this thesis and create a new array, with higher strength, in a few steps. The new array has another factor, has strength one higher than the original array, and has size Ns_{k+1}, where N is the size of the original array and s_{k+1} is the order of the new factor.

First we take the original array, and add the first element of the new factor as a last row. The element must be repeated N times, once for each column of the array. This process is then repeated for each element of the new factor, repeating each element N times. When all elements of the new factor have been used, all the copies of the original array, each with a new last row, are put side by side to form the new array.

We illustrate this using array (4.1), shown in Section 4.2, as our original array and using $\mathbb{Z}_2$ as the new factor. The original array has strength 2 and size $N = 12$.

For the first element of the new factor we have:

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

For the second element we have:

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

We now have an array of strength 3 and size $N = 24$:

$$\begin{pmatrix} e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c & e & x & y & a & b & c \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

We note that the array above is essentially identical to the $\frac{1}{2}$ fraction of $G = S_3 \times \mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_2$ constructed in Section 4.2. Let us call the latter array D. The last row of the array above is the second row of D, the second row of the array is the third row of D, and the third row of the array is the last row of D. In fact, this turns out to be true for all arrays constructed in this chapter.

This method of increasing strength also works for the arrays in Section 5.2. It is unknown if a method exists in other cases.

Chapter 5

Arrays Not of Minimum Size

In this section we consider five more non-abelian groups G. The construction method of Section 4.2 may be applied to create arrays of minimum size on G, but the counting function of the resulting arrays is not constant on conjugacy classes. By making small changes in the method, we create a suite of conjugacy arrays that are not of minimum size L_{k-1}, though in four cases they are still proper fractions. As noted in Section 2.1, the size N must be a multiple of L_{k-1}.

In Section 5.5 we also consider an example in which we use two non-abelian factors.

5.1 Construction Techniques

We again construct the array as a $k \times N$ matrix. Row i will contain the elements of G_i, repeated in certain patterns. We will adopt a fixed order for the elements of the groups used in construction. The fixed order for each group is shown below with the group S_3 having two orderings.

- $S_3 - e, x, y, a, b, c$ (1)
- $S_3 - e, a, b, c, x, y$ (2)
- $Dih_5 - e, a, d, b, c, q, w, x, y, z$
- $\mathbb{Z}_n - 0, 1, 2, \ldots, n-1$

Note that conjugate elements are adjacent in these orderings, as can be seen in (3.1).

5.2 Construction of a $\frac{1}{2}$ Conjugacy Array

In constructing this type of array we use the first ordering of S_3. The two examples constructed are $\frac{1}{2}$ fractions of $G = S_3 \times \mathbb{Z}_6 \times \mathbb{Z}_6$ and $G = S_3 \times \mathbb{Z}_6 \times \mathbb{Z}_6 \times \mathbb{Z}_6$. We note that the orders have a common factor of 6, yet the conjugacy array is constructed using the method in Section 4.2. The minimum size of these arrays would be $L_{k-1} = 36$ and $L_{k-1} = 216$, respectively, which would result in a $\frac{1}{6}$ fraction.

It is natural to consider defining

$$\begin{aligned} v_1 &= \frac{3L_{k-1}}{s_1} \\ v_2 &= \frac{3L_{k-1}}{s_2} \\ v_j &= \frac{3L_{k-1}}{s_2 \cdots s_j} \text{ for } j > 2. \end{aligned}$$

We note that the first v_k elements of G_1 form a union of conjugacy classes, and that the array constructed is a $v_k/s_1 = 1/2$ fraction.

5.3 Construction of a $\frac{2}{3}$ Conjugacy Array

We use the second ordering of S_3 with the groups $G = S_3 \times \mathbb{Z}_3 \times \mathbb{Z}_3$ and $G = S_3 \times \mathbb{Z}_3 \times \mathbb{Z}_3 \times \mathbb{Z}_3$. For these examples the s_i have a common factor of 3. We define

$$\begin{aligned} v_1 &= \frac{2L_{k-1}}{s_1} \\ v_2 &= \frac{2L_{k-1}}{s_2} \\ v_j &= \frac{2L_{k-1}}{s_2 \cdots s_j} \text{ for } j > 2. \end{aligned}$$

We note also that the first v_k elements of G_1 will form a union of conjugacy classes, and that the array constructed is a $v_k/s_1 = 2/3$ fraction. The minimum size of these arrays would be $L_{k-1} = 18$ and $L_{k-1} = 54$, respectively, which would result in a $\frac{1}{3}$ fraction.

The construction does not follow the same steps as the $\frac{1}{2}$ and $\frac{1}{4}$ fractions. To construct the first row of the matrix we write the elements of G_1 in the second fixed order. Next we write the elements of G_1 again, this time in the order reversed. We alternate between the fixed order and the order reversed until there are $2L_{k-1}$ elements.

The steps to assemble rows $2, \ldots, (k-1)$ are identical to those in Section 4.2.

A new pattern is necessary to assemble the kth row. Repeat each element of G_k v_k times. These elements are arranged in the fixed order. Then we repeat each element of G_k v_k times with the order cyclically permuted. This pattern is repeated as needed to fill the row.

We show this for $S_3 \times \mathbb{Z}_3 \times \mathbb{Z}_3$. The array is the juxtaposition of three blocks.

$$\begin{pmatrix} e & a & b & c & x & y & y & x & c & b & a & e & e & a & b & c & x & y & y & x & c & b & a & e & e & a & b & c & x & y & y & x & c & b & a & e \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 0 & 0 & 0 & 0 \end{pmatrix}$$

It would be of interest to use this method in Section 5.2, where the orders have common factor 6, to see if a conjugacy array can be formed.

5.4 Smallest Array is the Complete Design

For the example $G = Dih_5 \times \mathbb{Z}_5 \times \mathbb{Z}_5$ the common factor of the orders is 5. We might expect the conjugacy array to be $\frac{1}{5}$ the size of the complete design. When attempting to construct this example using the method described in Section 4.2 we find that the array is not constant on conjugacy classes. It is natural to attempt other multiples of L_2 for the size of the array. We obtain a conjugacy array when we apply that method with:

$$\begin{aligned} v_1 &= \frac{5L_2}{s_1} \\ v_2 &= \frac{5L_2}{s_2} \\ v_3 &= \frac{5L_2}{s_2 s_3}. \end{aligned}$$

This conjugacy array is the size of the complete design, $5L_2$. For this example $L_2 = 50$, so $5L_2 = 250$, which is the size of the complete design and therefore a trivial result.

5.5 Example with Two Non-Abelian Factors

In the process of constructing arrays for this thesis the question was raised, what would happen in an array constructed with more than one non-abelian factor? To investigate we try one $6 \times 6 \times 6$ example with two non-abelian factors.

The group we use is $G = S_3 \times S_3 \times \mathbb{Z}_6$, rather than $G = S_3 \times \mathbb{Z}_6 \times \mathbb{Z}_6$ used in Section 5.2. We see that $S_3 \times S_3$ has more complicated conjugacy classes than $S_3 \times \mathbb{Z}_6$, which left us unsure where to start.

We started by making the following substitutions in the second row of the array in Section 5.2 from $\mathbb{Z}_6$ to S_3:

$$0 \to e$$

$$1 \to x$$

$$2 \to y$$

$$3 \to a$$

$$4 \to b$$

$$5 \to c$$

The resulting array is essentially identical to the array constructed using the group $G = S_3 \times \mathbb{Z}_6 \times \mathbb{Z}_6$.

The array on $G = S_3 \times S_3 \times \mathbb{Z}_6$ has strength 2 and, perhaps surprisingly, its counting function is still constant on conjugacy classes. It is unknown if other examples would result in a conjugacy array using this substitution method.

Chapter 6

Conclusion

Table 6.1 shows constructions created for this thesis using one non-abelian factor, where $k-1$ is the strength, L_{k-1} is defined by (2.4), and the size of the array is L_{k-1} unless otherwise noted, and the fraction is the size of the array compared to that of the complete array.

In this thesis every array has strength $k-1$ and its counting function is constant on conjugacy classes, but the size and the groups vary. We have seen that a common factor of the orders s_i plays a role in determining the size of the array and the fraction of the complete design. We would like to clarify what exactly that role is. The algorithm we have is essentially identical in all cases, but some steps do change depending on the common factor of the orders s_i. Would other algorithms result in minimum sized solutions?

As we have seen in one example, a substitution can be made such that the group G would have two non-abelian factors and we still have a conjugacy array. One would assume this substitution will not result in a conjugacy array for many examples, or for a group with many non-abelian factors. This is because the condition that the array be constant on conjugacy classes makes constructions more difficult. It is left for further investigation to see if the method of substituting a non-abelian factor in for an abelian one will result in a conjugacy array and if this process can be generalized.

Table 6.1: Arrays Constructed In This Thesis

Complete Design	Size of Complete Design	Strength $k-1$	L_{k-1}	Size of Array		Fraction
$6 \times 2 \times 2*$	24	2	12	12		1/2
$6 \times 2 \times 2 \times 2*$	48	3	24	24		1/2
$6 \times 3 \times 3$	54	2	18	36	$= 2L_{k-1}$	2/3
$6 \times 3 \times 3 \times 3$	162	3	54	108	$= 2L_{k-1}$	2/3
$6 \times 4 \times 4$	96	2	48	48		1/2
$6 \times 4 \times 4 \times 4$	384	3	192	192		1/2
$6 \times 6 \times 6$	216	2	36	108	$= 3L_{k-1}$	1/2
$6 \times 6 \times 6 \times 6$	1,296	3	216	648	$= 3L_{k-1}$	1/2
$6 \times 4 \times 2$	48	2	24	24		1/2
$6 \times 6 \times 2$	72	2	36	36		1/2
$6 \times 6 \times 4$	144	2	72	72		1/2
$8 \times 2 \times 2$	32	2	16	16		1/2
$8 \times 2 \times 2 \times 2$	64	3	32	32		1/2
$8 \times 2 \times 2 \times 2 \times 2$	128	4	64	64		1/2
$8 \times 2 \times 2 \times 2 \times 2 \times 2$	256	5	128	128		1/2
$8 \times 4 \times 4$	128	2	32	32		1/4
$8 \times 4 \times 4 \times 4$	512	3	128	128		1/4
$8 \times 6 \times 6$	288	2	144	144		1/2
$8 \times 6 \times 6 \times 6$	1,728	3	864	864		1/2
$8 \times 4 \times 2$	64	2	32	32		1/2
$8 \times 6 \times 2$	96	2	48	48		1/2
$8 \times 6 \times 4$	192	2	96	96		1/2
$10 \times 2 \times 2$	40	2	20	20		1/2
$10 \times 2 \times 2 \times 2$	80	3	40	40		1/2
$10 \times 4 \times 4$	160	2	80	80		1/2
$10 \times 4 \times 4 \times 4$	640	3	320	320		1/2
$10 \times 6 \times 6$	360	2	180	180		1/2
$10 \times 6 \times 6 \times 6$	2,160	3	1,080	1,080		1/2
$10 \times 4 \times 2$	80	2	40	40		1/2
$10 \times 6 \times 2$	120	2	60	60		1/2
$10 \times 6 \times 4$	240	2	120	120		1/2

* Another example is constructed in [1].

Bibliography

[1] Jay H Beder and Jesse S Beder. Generalized wordlength patterns and group structure. *Journal of Statistical Planning and Inference*, 2012. To appear.

[2] Hongquan Xu and C. F. J. Wu. Generalized minimum aberration for asymmetrical fractional factorial designs. *The Annals of Statistics*, 29:1066–1077, 2001.

APPENDIX A: Minitab Procedure for Verifying Strength

1. Enter the rows of the array as columns in a Minitab worksheet. The columns may be copied from Excel. (Note that the Maple procedure in Appendix B will read this information from Excel.)

2. Change the numeric values in the columns to text. (Go to Data >Change Data Type)

3. Label each column as A, B, C, . . . to distinguish between each factor of the array. Call these **factor columns**.

4. Create k **combination columns** by concatenating each set of $k-1$ factor columns. (Data >Concatenate). Label these columns. For example, for factor columns A, B, C, and $k=2$, we have AB, AC, and BC.

5. Tally the entries in each of the combination columns (Stat >Tables >Tally Individual Variables). Check that each entry in a given column occurs the same number of times. This verifies that the strength is $k-1$.

APPENDIX B: Maple Code to Verify the Conjugacy Condition

```
>restart;
```

Load the necessary Maple packages.

```
>with(combinat):
with(Spread):
with(linalg)
```

The array should be stored in Excel, the rows of the array stored as columns in Excel. Save the Excel file as a text file. Load the text file by typing in the name of the saved text file. Indicate the number of columns.

```
>Mdata := readdata("S3 x S3 x 6.txt", string, 3):
```

Convert the array in the text file to a matrix in Maple.

```
>M := Matrix(nops(Mdata),3, (i,j) -> convert(M[i][j], symbol));
```

Enter code to create a list whose elements are the elements of the cartesian product of a list, L, of sets.

```
>CPL := proc(L)
    local cpf, ans: ans:=NULL:
    cpf := cartprod(L):
    while not cpf[finished] do
            ans := ans, cpf[nextvalue]()
    od; [ans];
end
```

Enter code to create a sequence of elements using CPS, which is similar to CPL. When the output is surrounded by {} repeats are omitted.

```
CPS:= proc(L)
```

```
    local cpf, ans: ans:=NULL:
    cpf:= cartprod(L):
    while not cpf[finished] do
            ans:= ans, cpf[nextvalue]()
    od; ans;
end
```

Enter code to find the index i such that C[i] = elmt.

```
lookup := proc(elmt)
    local i, ans:
    for i from 1 to 24 do
            if C[i]=elmt then ans := i fi
    od: ans; end
```

Enter code to take a list LST and apply CPS to each element of CPL(LST).

```
fast_PART:= LST -> map(CPS, CPL(LST)):
```

Enter code to create a function from ARY that has value at x the number of y in ARY equal to x.

```
    f := y -> x -> if x=y then true else false fi:
    mult :=ARY -> y -> nops( select(f(y), ARY) ):
```

Enter code that determines if the array is constant on conjugacy classes.

```
GOOD := ARY ->
    if nops({seq(nops( map( mult(ARY), PART[j] )),j=1..nops (PART))})=1
    then true else false
fi:
>?file
```

This is where you start to enter the group. The group is called C. Enter the elements of each factor. Then all elements are generated as a Cartesian product.

```
>C := CPL( [[e,x,y,a,b,c], [0,1,2,3,4,5], [0,1]] );
```

*PART is a *global* variable that must be entered by a list of partitions of factors (def. by conj. class). Enter the conjugacy classes of each factor. Generate a list of all partitions.*

```
>PART := fast_PART(
    [
{{a,b,c}, {x,y}, {e}},
{{0}, {1}, {2}, {3}, {4}, {5}},
{{0}, {1}}
]);
```

Find the number of partitions.

```
>nops(PART);
```

Create a spreadsheet from the text file.

```
>ssid:= CreateSpreadsheet();
```

Specify the size of the spreadsheet. Find the upper left cell and enter the row and column separated by commas. (Upper left cell will be 1, 1) Next find the lower right cell and enter the row and column separated by commas.

```
>SetSelection(ssid, 1, 1, 36, 3);
```

Create a matrix, A, out of the columns in the spreadsheet.

```
>A:= GetValuesMatrix(ssid);
```

List the rows of the array from the matrix. Indicate the number of rows in the array.

```
>A_list:= [seq( convert(row(A,i), list), i=1..36 )];
```

Determine if the array is constant on conjugacy classes.

```
>GOOD(A_list);
```

Upper case commands are used for more complex combinatorial data.