Response Surface Methodology for Split-Plot Designs with Categorical Factors

by

Jenna Gaylene Tichon

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfilment of the requirements of the degree of

Doctor of Philosophy

Department of Statistics
University of Manitoba
Winnipeg

Abstract

Split-plot designs often arise in agriculture and industrial experimentation when some factors are harder to vary than others, leading to randomization restrictions. This has an effect on both the run order and analysis of the experiment.

Response surface methodology (RSM) split-plot designs for experiments with quantitative factors have received a lot of coverage in the literature. These designs are not appropriate, however, if categorical factors are also present.

Draper and John (1988) and Wu and Ding (1998) explore techniques for adding categorical factors to non-split-plot RSM designs. Building on their initial idea of adding the categorical factor sequentially after taking an initial base design in the quantitative factors, this thesis explores how to add a two-level categorical factor in the split-plot RSM setting.

Due to the randomization restrictions, adding a categorical factor in the split-plot setting requires considerably more care, in order to meet basic feasibility requirements and to maintain the structure. We explore four techniques for adding categorical factors and present results on requirements for the feasibility of proposed assignments of the categorical factor. We find that not all methods are appropriate for every base design.

Throughout the thesis, we expand upon an example of an RSM split-plot experiment in quantitative factors for a ceramic pipe experiment from Vining and Kowalski (2008), by introducing a hypothetical additional categorical factor at either

the whole-plot (hard-to-vary) or split-plot (easy-to-vary) level. We discuss optimal strategies for assigning a factor, conduct some initial exploration of the different response surfaces after perturbations to the data using contour plots, and suggest further avenues for analysis.

The thesis culminates in tables of D-optimal designs for the various assignment methods based on an algorithm and computer code written for the various assignment methods.

Acknowledgment Page

I first need to thank my supervisor, Dr. John Brewster, for being a wonderful mentor to me both academically and professionally. His support and dedication to seeing me through my graduate program, even well after his retirement, has helped my work immeasurably. Thank you as well to my thesis committee, Dr. Saumen Mandal, Dr. Po Yang, Dr. Mohamed Bassuoni, and Dr. Tim Robinson, for their insightful feedback and helpful suggestions. My thesis work was also partially supported through an NSERC postgraduate scholarship.

I would like to thank all of the support staff in the department for all of their guidance and help navigating all of the requirements in the department and at the university. In particular, I would like to acknowledge Dave Gabrielson for all of his help using the computational cluster and the many, many, typesetting and formatting questions that he fielded throughout my career.

While I have dedicated this thesis to all of my teachers, I would like to personally thank Mr. Art Penning, Mr. Al Allison, and Dr. Terry Visentin for sparking my love of math, offering unfailing belief in me, and being mentors to me at different stages in my academic career when I needed it most.

Lastly, but most importantly, thank you to my parents, my friends, my family, and my partner Freddie for seeing me through my schooling and being there with a hug when things were hard and with the best cheers at every milestone. I could not have done it without every single one of you.

Dedication Page

To all of my teachers, for always seeing the best in me and showing me a world of numbers that is more beautiful than I could have ever imagined.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Response Surface Methodology

In response surface methodology, we would like to fit a surface for a response variable we believe is a function of several variables x_1, x_2,...,x_k and a random error term. Three potential goals are: we are interested in the actual model of the response surface, we want to know which factor settings maximize or minimize the response variable, or we want to know the actual value of the minimum or maximum of the response surface.

For two quantitative factors, x_1 and x_2, we could fit a first-order model as given by:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon, \tag{1.1}$$

and possibly add an x_1x_2 term to fit a first-order model with interaction.

If there is curvature in the response surface, we could fit a second-order model as given by:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \epsilon. \tag{1.2}$$

It is likely that model 1.2 will not be appropriate over a very wide region but it should be a useful approximation around the maximum or minimum.

When fitting a first-order model, as in equation 1.1, it is common to run a 2^2 full factorial design in coded units, as in Figure 1.1. Adding $c \geq 2$ center points, assuming homogeneous variance, allows us to also estimate the error variance and investigate lack of fit.

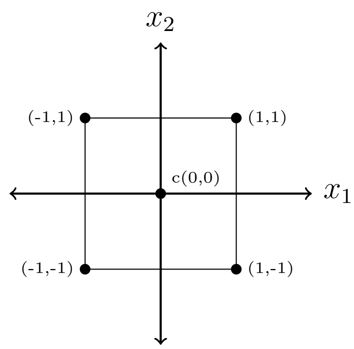

FIGURE 1.1. A 2^2 full factorial design in x_1 and x_2, plus c center points.

Figure 1.1, excluding the center points, shows a design that has all combinations of 2 factor levels for 2 quantitative factors. A design with all combinations of 2 factor levels for k factors is called a 2^k full factorial design. If the number of factors makes running a full factorial design too large to be practical, a 2^{k-p} fractional factorial (FF) design is often run instead. These designs have k factors but p of them are generated by multiplying other factors together. They have 2^{k-p} runs.

Full factorial and fractional factorial designs are often run as preliminary screening experiments to narrow down the number of factors that are of likely interest. After testing for the significant effects, these experiments are often followed up with response surface designs using the important factors.

For fitting the second-order model, one of the most common designs is a central composite design (CCD). For the case of two quantitative variables, a CCD is shown in Figure 1.2. The design points include all $2^2 = 4$ factorial points, c center runs and four axial points at $(0, \pm\sqrt{2})$ and $(\pm\sqrt{2}, 0)$.

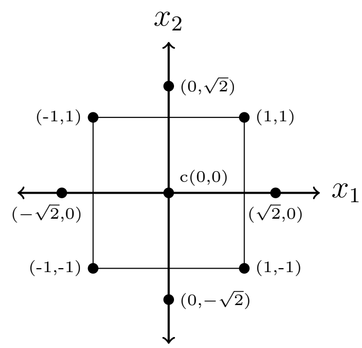

FIGURE 1.2. A central composite design in x_1 and x_2 used for fitting a second-order model. With $\alpha = \sqrt{2}$, this design is both rotatable and spherical.

The addition of axial points allows for the estimation of all the second-order terms in the model. For the general CCD, axial points at $(\pm\sqrt{\alpha}, 0, .., 0)$, $(0, \pm\sqrt{\alpha}, 0, \ldots, 0)$,...,$(0, 0, \ldots, 0, \pm\sqrt{\alpha})$, are often chosen in such a way to make the design rotatable. This means that all points at an equal distance from the center have the same variance in the predicted response. In a CCD, for general k, this is given by

$$\alpha = (n_F)^{1/4} \tag{1.3}$$

where n_F is the number of factorial points in the design (Montgomery, 2012).

With two factors, as in Figure 1.2, there are four factorial points so that $\alpha = (4)^{1/4} = \sqrt{2}$. In an experiment with two factors, choosing $\alpha = \sqrt{2}$ also makes the design spherical.

We can estimate the coefficients for the fitted model using least-squares, then use contour plots or calculus to find the optimum (if such an optimum exists).

For example, if the fitted model is given by

$$\hat{y} = 2 + 4x_1 + 3x_2 + 3x_1^2 + 4x_2^2 + x_1x_2, \tag{1.4}$$

we can look at the contour plot made in R in Figure 1.3 to approximate the location of the minimizing value. As the inner ellipse has the smallest height, the minimizing value will be found in that region.

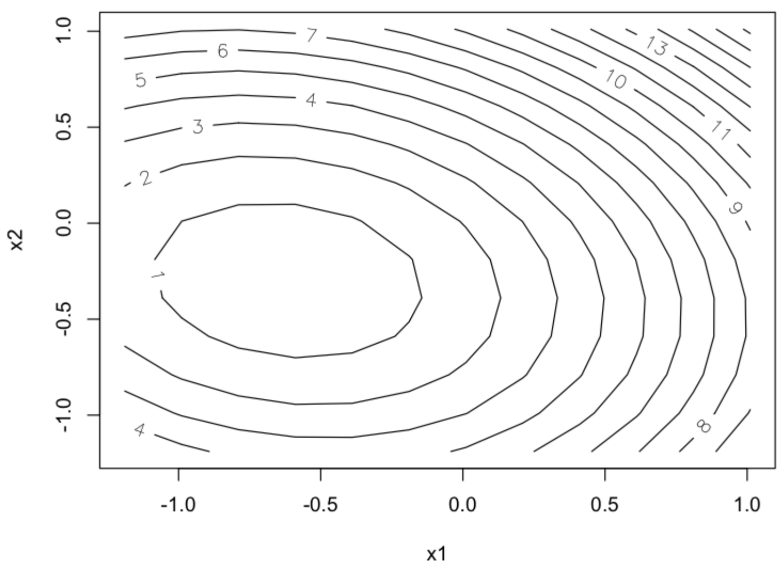

FIGURE 1.3. Contour plot for the response surface of the fitted model $\hat{y} = 2 + 4x_1 + 3x_2 + x_1x_2 + 3x_1^2 + 4x_2^2$. The labels indicate a minimum in the response surface in the inner ellipse.

Alternatively, we could use calculus to find the minimum by taking the partial derivatives of equation 1.4 and setting them to zero. This gives the following system of equations:

$$\begin{aligned} \frac{\partial \hat{y}}{\partial x_1} &= 4 + 6x_1 + x_2 &= 0 \\ \frac{\partial \hat{y}}{\partial x_2} &= 3 + 8x_2 + x_1 &= 0. \end{aligned} \tag{1.5}$$

By solving these equations, we find the exact minimum is achieved at the point $(-5/7, -2/7)$.

1.2 Response Surface Methodology with Categorical Factors

A complication arises in design selection when one or more of the factors is categorical. We will denote the categorical factors by z_1, z_2, ... , z_r, where r is the number of categorical factors. In such a situation, there is an obvious issue with using a standard CCD; namely, there may not be five levels of the categorical factor. For example, there may only be two different brands of adhesive that are of interest. Furthermore, if by chance there were five different levels, there likely would not be a logical ordering or sense of what the "center" factor level should be. You are tied to using the levels as they are given to you.

In the simplest case, consider an experiment with one categorical and one quantitative factor. A logical design would be to run various levels of the quantitative factor at each level of the categorical factor.

A more interesting case arises with two quantitative factors and a third categorical factor. Suppose we wanted to fit a second-order model where the coefficients depend on z_1. In this case, the most straightforward option would be to run a full CCD in the quantitative factors at each level of the categorical factor. While it would allow us to fit a response surface at each level of the categorical factor, it is often too costly to double the size of the experiment. In settings where runs are costly, this option is infeasible. A different approach is required when we are limited to a certain number of runs.

For fitting first-order models, Draper and John (1988) examine how to construct a design by starting with a factorial design plus center points in the quantitative factors. Then those points are partitioned between the two levels of the categorical factor. Let us start with the small case with two quantitative random variables and one qualitative random variable. Here we begin with $2^2 = 4$ factorial points plus the center run for a total of five runs with four degrees of freedom. This allows us to estimate the coefficients for x_1, x_2, z_1, and potentially one other effect. The points should be partitioned such that the desired additional factor effect has a column that is linearly independent from the others in the design matrix and is clearly estimable.

Similarly, for fitting the second-order model, one could start with a standard CCD in the quantitative variables and then partition the points amongst the levels of the qualitative factor. A possible partition for the design points is given in Figure 1.4

However, there are complications. One issue with the partition in Figure 1.4 is the lack of a center run at each level of the categorical factor. Another issue is the sparseness of points for $z_1 = 1$. If we believe that the factor z_1 is important, we will be unable to estimate many effects at each level of z_1.

There are some improvements that could be made on this partition. As it is common for factorial designs to have multiple center points, we could partition several center points between the levels of z_1. If z_1 has an additive effect, we could

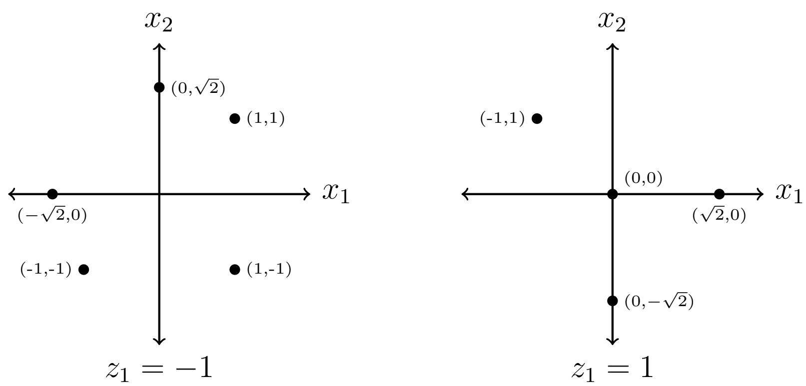

FIGURE 1.4. A central composite design partitioned between two levels of a third qualitative factor z_1 in Draper and John (1988). This design has a number of deficiencies.

estimate the effect by having one center point at each level of z_1. If we would like an estimate of variability and believe the error variance is homogeneous between levels of z_1, we could put two center points at one level of z_1. Ideally, if it is feasible within economic and time constraints, we could further have two center points at each level of z_1.

While there are some issues that will never be resolved in such a small experiment due to the minimal number of points we are partitioning, it is possible to improve what can be estimated. This is accomplished by partitioning the points more evenly between the two levels of z_1 such as in Figure 1.5.

To help deal with the issues arising from a strict partition, Draper and John (1988) advocate instead that the full factorial design with center points be run at each level of z_1 and to only partition the axial runs, as in Figure 1.6. This is not a partition in the strict sense, but could be considered as a partition with replication.

We can see that, when collapsed over z_1, we have a CCD and can fit the full second-order model as given by the equation.

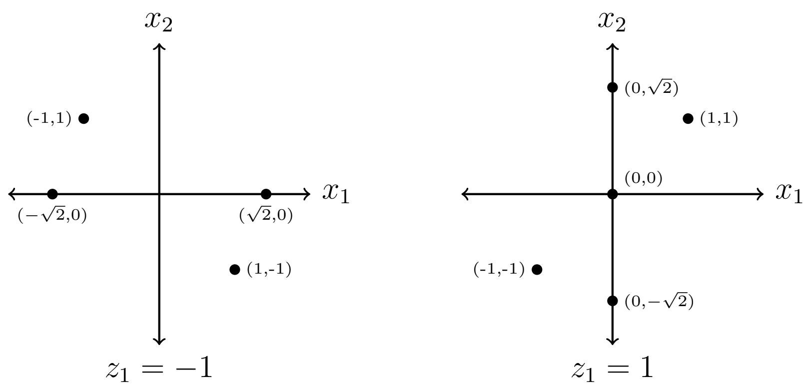

FIGURE 1.5. Another central composite design in quantitative factors x_1 and x_2. This design from Draper and John (1988) is somewhat better than the design in Figure 1.4 as the points are more evenly distributed amongst the levels of z_1.

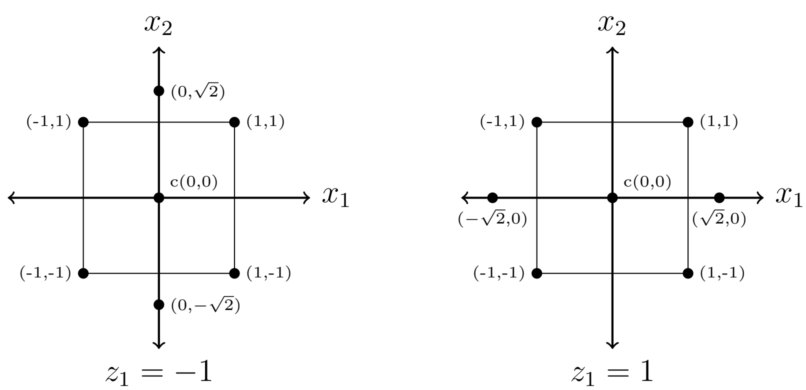

FIGURE 1.6. A full factorial design with center points in two quantitative factors x_1 and x_2 at each level of categorical factor z_1 with axial points from the quantitative factors partitioned between the levels of z_1. This is a superior design to that of Figure 1.5.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 z_1 + \beta_{13} x_1 z_1 + \beta_{23} x_2 z_1 + \beta_{12} x_1 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \epsilon \tag{1.6}$$

This has common second-order terms for both levels of z_1. This is a valuable feature as, if testing reveals the categorical factor to have an insignificant effect, there is a good design in the quantitative variables alone after collapsing over z_1.

We can choose which design to use depending on the additional effect(s) we would like to estimate. Any of these partitions with replication should be able to estimate the coefficients of all factors in equation 1.6. However, it is possible to estimate different numbers of additional terms ($x_1^2z_1$, $x_2^2z_1$, and $x_1x_2z_1$) based on the columns that are linearly independent in the design matrix. Our design in Figure 1.6, has a linear dependence amongst terms that are likely of interest in the experiment. Draper and John (1988) demonstrate that we will have the dependence $x_1^2 - x_2^2 = x_2^2z_1 - x_1^2z_1$, making only one of $x_1^2z_1$ or $x_2^2z_1$ estimable, whereas the design in Figure 1.7, for example, allows both of those terms to be estimated. We can differentiate between designs based on which of these additional terms we think may be significant and potentially added to the original model.

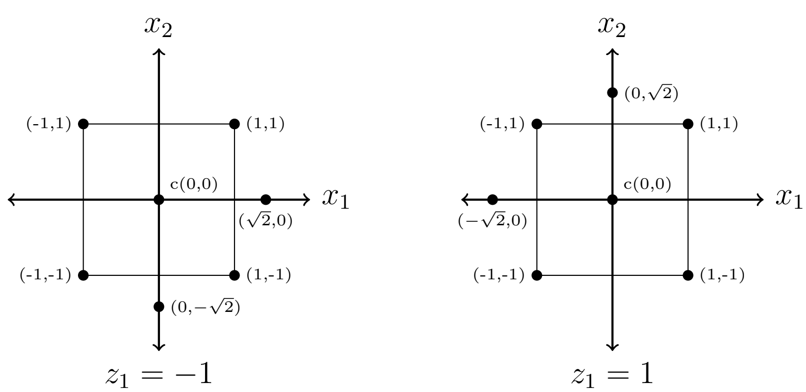

FIGURE 1.7. Another full factorial design with center points in two quantitative factors x_1 and x_2, at each level of categorical factor z_1, with axial points from the quantitative factors partitioned between the levels of z_1. This is a superior design to that of Figure 1.6 as the x^2 coefficients are all estimable.

Should we need to select between good partitions that estimate the same number of effects, we may rely on a letter optimality criterion to choose a final model. This will be explored later in the thesis where we choose to use D-optimality.

1.3 Response Surface Methodology for Split-Plot Designs

A second complication in RSM arises when running a split-plot experiment. In a split-plot experiment there are two types of factors: whole-plot factors and split-plot (or sub-plot) factors. We will consider designs where we have r whole-plot factors and s split-plot factors.

Whole-plot (WP) factors are harder to vary due to constraints such as expense, time, or physical limitations. As these factors take a long time to reset, or are expensive to continually reset, it may be desirable to keep the factor level constant for several runs at a time. This structure leads to randomization restrictions. Split-plot (SP) factors are relatively easy to vary and can be changed with less hardship from trial to trial.

In Table 1.1 we can see an example of the factorial portion of a split-plot experiment with one whole-plot factor and two split-plot factors. Although this design is too small for practical purposes, it serves to illustrate the structure. We will denote whole-plot factors with capital X's and split-plot factors with lower case x's. In the first and last four runs, the factor X_1 is held constant while the split-plot factors vary. In this notation, we do not repeat the "-1" value for each X_1, for each of the 4 runs in this whole plot, to make it easier to see the split-plot structure. To conduct this experiment, a practitioner would randomly select one of those two sets of four (also known as a whole plots) to run first and then randomly select the order for the four runs within that set.

To fit a model for a split-plot design in quantitative factors using RSM, one

TABLE 1.1. A split-plot experiment with one WP and two SP factors

X_1	x_1	x_2
-1	-1	-1
	-1	1
	1	-1
	1	1
1	-1	-1
	-1	1
	1	-1
	1	1

would also add center points and axial points for the various factors. The way to do this, however, is not always clear. There are complications such as whether or not it is possible to have five levels of a factor that is truly hard to vary. This is one reason for using "face-centered" designs with three levels, as illustrated below. Adding axial runs for each factor can affect the experiment by adding too many whole plots or could result in whole plots that are needlessly large. There are different ways to assign the additional runs in the literature which will be explored further in the thesis.

In Vining and Kowalski (2008), the authors give an example of a split-plot design used for response surface methodology and show how to estimate the model terms. In this experiment, an engineer is studying the effect of four different factors on the strength of a ceramic pipe being cured in a furnace. The four factors are:

- temperature in zone 1 of furnace,
- temperature in zone 2 of furnace,
- amount of binder in the formulation,

- grinding speed of the batch.

As the temperatures of the two zones in the furnace take a long time to reset, they are harder to vary than the binder formulation or the grinding speed. For this reason, it is desirable to maintain the temperatures in the two zones of the furnace for multiple runs at a time. As such, they are designated as whole-plot factors. As the binder formulation and grinding speed can be changed easily, they are designated as split-plot factors and their settings will be reset each run.

The design used for this experiment, with the corresponding values of the response variable, can be found in Table 1.2. We see that it is a 48-run design conducted in 12 whole plots of size four. There are 16 factorial points, 12 center points, and 20 axial points. Note that the axial points are "face-centered", being on the boundaries of the factorial region. This means that only three levels of the whole-plot factors are used.

This design is run by first randomly selecting a whole plot and running all four runs within the whole plot in random order. This keeps the values of the whole-plot factors (X_1 and X_2) constant for four runs at a time, to help with the difficulty of resetting those factors. After completing a whole plot, another whole plot is randomly selected and run in a similar manner.

Through the use of RSM, Vining and Kowalski (2008) fit the linear model

$$\begin{aligned}\hat{y} = \quad & 74.9055 + 4.5579X_1 - 6.5592X_2 + 1.7381X_1^2 - 0.5407X_2^2 + 0.8431X_1X_2 \\ & -4.9730x_1 + 4.0922x_2 - 2.3864x_1^2 + 2.5736x_2^2 - 1.0394x_1x_2 + 1.4356X_1x_1 \\ & -1.4794X_1x_2 - 1.0019X_2x_1 + 1.9856X_2x_2. \end{aligned} \tag{1.7}$$

X_1	X_2	x_1	x_2	y	X_1	X_2	x_1	x_2	y
-1	-1	-1	-1	80.40	0	-1	0	0	80.07
		1	-1	71.88			0	0	80.79
		-1	1	89.91			0	0	80.20
		1	1	76.87			0	0	79.95
1	-1	-1	-1	87.48	0	1	0	0	68.98
		1	-1	84.49			0	0	68.64
		-1	1	90.84			0	0	69.24
		1	1	83.61			0	0	69.20
-1	1	-1	-1	62.99	0	0	-1	0	78.56
		1	-1	49.95			1	0	68.63
		-1	1	79.91			0	-1	74.59
		1	1	63.23			0	1	82.52
1	1	-1	-1	73.06	0	0	0	0	74.86
		1	-1	66.13			0	0	74.22
		-1	1	84.45			0	0	74.06
		1	1	73.29			0	0	74.82
-1	0	0	0	71.87	0	0	0	0	73.60
		0	0	71.53			0	0	73.59
		0	0	72.08			0	0	73.34
		0	0	71.58			0	0	73.36
1	0	0	0	82.34	0	0	0	0	75.52
		0	0	82.20			0	0	74.74
		0	0	81.85			0	0	75.00
		0	0	81.85			0	0	74.90

TABLE 1.2. The design for the ceramic piper experiment, with two hard-to-vary and two easy-to-vary factors, together with the data arising from the experiment.

In the analysis, all effects except x_1^2, x_2^2, and x_1x_2 show as statistically significant, with p-values close to zero. While not done explicitly in Vining and Kowalski, this suggests it would be reasonable to test if the second-order whole-plot terms may not be necessary and by testing the complete model against a reduced model to see if these higher order terms should be dropped.

Designs for fitting second-order split-plot models with quantitative factors will be thoroughly covered in chapter 2. For now, we will look what happens if we were to add a complication to our experiment, a categorical factor.

1.4 Split-Plot Response Surface Designs with Categorical Factors

While response surface methodology and split-plot designs are commonly discussed on their own in the literature, questions arise when we have an experiment that is both split-plot in nature and has categorical factors. If we are assigning the categorical factor by partitioning the whole design, there is a question of how this can be done optimally while still preserving the split-plot structure.

In our ceramic pipe example, we can imagine a different setup where there are two substantially different industry standard furnaces that might be used to cure the ceramic pipes. The factor is categorical in nature and only has two levels: -1 and 1. As temperature stays constant for several runs at a time, the type of furnace itself must be a whole-plot factor as well.

The question is, when I add the column for the categorical factor to the design in Table 1.2 , how do I assign the levels for the new categorical factor, type of furnace?

If we are adding the hard-to-vary factor, type of furnace, we cannot assign the points completely at random as the whole plots will likely start to vary too often if both -1 and 1 are assigned in the same whole plot.

Instead, we may need to add a split-plot categorical factor, such as the positioning of the pipe in the oven. As the pipes are placed by us, we can change them from run to run. If we are adding the easy-to-vary factor, pipe position, we also cannot assign the values for z_1 in the design without regard to structure either. While these factors are allowed to vary from run to run and within the same whole plot as they are easy-to-change, there is another concern. Generally speaking, we would like to preserve balance (have an equal number or nearly equal number of -1's and $+1$'s) in the split-plot factors within each whole plot. If we assign the level of a categorical split-plot factor without paying attention to this structure, we may get entire whole plots with all the same value of pipe position and then it is not truly a split-plot factor any more.

Thus, whether a categorical factor is a whole-plot or a split-plot factor will also play a role in our decision for how to partition the experiment as they both have different concerns. Both types of factors follow different rules for how they are required to vary and need to be assigned accordingly.

In Chapter 3, we will examine four different methods for assigning the levels of a categorical factor that maintain the "spirit" of the method in Draper and John (1988) for completely randomized designs. That is, we will start with a good base design in the quantitative factors and consider assigning the categorical factor through some combination of partitioning and replication. For whole-plot categorical factors we will look at three different methods: doubling the design, splitting the whole plots, and a method we will call pseudo-partitioning. For splitting the whole plots we will literally divide the base design's whole plots into two smaller whole plots. For pseudo-partitioning we will employ a combination of partitioning and replication to maintain the whole plot size but have some increase in the size of the overall design. The assignment of our factors furnace type or pipe position will be looked at in Chapter 3. In Chapter 4, we will briefly visit the issue of fitting the model where we would like to fit separate response surfaces for each furnace type or pipe position as well as future areas of research.

In the thesis we will discover that not all of these methods are appropriate or feasible for every base design. We will see that many of these methods can be produced by using generating factors and learning the conditions under which we can use generation. For designs that can use multiple methods, we will compare different assignments of the categorical factor with practical advice on choosing between them.

After explaining the methods, we will discuss a computer search algorithm that I wrote to help select optimal designs. It works based on inputting a good base design, selecting the assignment method and then cycling through all possible assignments of the categorical factor. The algorithm tests which assignments are feasible (i.e. which lead to an assignment that preserves the split-plot structure and maintains certain criteria) and then uses an optimality criterion to find the globally optimum assignment of the categorical factor. In this thesis we consider the D-optimality criterion, though others could be incorporated as well.

This algorithm will culminate in the creation of tables of optimal RSM split-plot designs with categorical factors for each of the methods. The base designs will be cultivated from commonly used designs in the literature that have available tables of optimal designs.

Chapter 2

Literature Review

2.1 Response Surface Methodology with the Addition of Categorical Factors

When designing an experiment, it may happen that there will be qualitative factors as well as quantitative factors. For example, you might have several different options for a type of wood to use or there could be different brands of a battery. When trying to use a second order model in RSM, there are several issues that need to be addressed due to the nature of the qualitative factors.

Many articles relating to categorical RSM such as Draper and John (1998), and Sarika (2005) point out the most obvious set back that arises with qualitative factors, namely that we lack an ability to have center runs and axial points. Firstly, it may not be an option to have five levels of the categorical factor if they do not exist. More importantly, as there is no continuity in the levels, any assignment of a "center run" would be arbitrary and not useful in determining a second-order model and lack of fit.

The most straightforward approach to dealing with this issue would be to run an entire CCD at each level of the qualitative factor. While this would give you the best view of the response surface at every level, it would would not be a very economical nor would it be a practical approach as it could require a very large amount of runs, particularly if there were many levels of the categorical factor or the number of quantitative factors is large. As will be stressed many times throughout the thesis, in industrial experimentation, we are limited by both time and money which must constantly be kept in mind when choosing our designs. A more economical approach in terms of run size suggested by Draper and John (1988) and Wu and Ding (1998) is to find a design appropriate for RSM that would be considered "good" without the qualitative factor and then assign each run to a different level of the qualitative factor according to a predesignated criterion.

Draper and John advocate that this be done in a way such that we can estimate all parameters of interest in the model, test for goodness of fit when adding interaction terms between qualitative and quantitative factors, test for goodness of fit for higher-order terms in the quantitative factors and ensure that there are a reasonable number of design points. As well, any design must meet the following conditions:

- If our complete focus is on fitting a first-order model, there is at least one point at each level of z and, when we collapse over z, we have a design that allows us to fit the full first-order model.
- If our focus is on fitting a second-order model, when we collapse over z there must be a design with enough points to fit a full second-order model. Each level of z must have enough points to clearly estimate a full first-order model.

Consider an experiment with k different quantitative variables denoted by x_i for $i = 1, \ldots, k$ and one qualitative random variable z_1 that has m levels. We may decide to use one of the following models depending on how we believe the qualitative factor might interact with the quantitative factor and whether or not we are considering a first- or second-order model.

If we believe that the response surface is identical at each level of the qualitative factor, with the exception of a possible vertical shifting parameter to create parallel surfaces, we will get either the first-order model in equation 2.1,

$$E(Y) = \sum_{z_1} W_{z_1} \left(\beta_0(z_1) + \sum_{i=1}^{k} \beta_i x_i \right), \tag{2.1}$$

or the second-order model in equation 2.2,

$$E(Y) = \sum_{z_1} W_{z_1} \left(\beta_0(z_1) + \sum_{i=1}^{k} \beta_i x_i \right) + \sum_{i<j}^{k} \beta_{ij} x_i x_j + \sum_{i=1}^{k} \beta_{ii} x_i^2. \tag{2.2}$$

In these models we are implying that z_1 has a simple additive effect. All that will change is the intercept term depending on the level of z_1.

The notation is a slightly modified version of the style of Wu and Ding (1998). The W_{z_1} acts as an indicator to decide which is the appropriate response surface based on the level of the qualitative factor. Hence W_{z_1} is equal to 1 if Y is at level z_1 of the qualitative factor and is equal to zero otherwise. Then $\beta_0(z_1)$ is the intercept when the qualitative random variable is at level z_i and similarly with the slope term

$\beta_i(z_1)$ for the i^{th} quantitative factor.

Note that in equations 2.1 and 2.2, only the intercept term changes based on the level of the qualitative factor. This is what indicates that it is solely a vertical shift.

In the case where we believe that there may be interaction with z_1, and non-parallel surfaces, but with common second-order terms, for the full second-order model we get equation 2.3

$$E(Y) = \sum_{z_1} W_{z_1} \left(\beta_0(z_1) + \sum_{i=1}^{k} \beta_i(z_1) x_i \right) + \sum_{i<j}^{k} \beta_{ij} x_i x_j + \sum_{i=1}^{k} \beta_{ii} x_i^2. \tag{2.3}$$

It is also possible to view the model in the form of a departure from the mean model as

$$E(Y) = \beta_0 + \sum_{i=1}^{k} \beta_i x_i + \sum_{i=1}^{k} \sum_{j=1}^{k} \beta_{ij} x_i x_j + \sum_{z_1} W_{z_1} \gamma_0(z_1) + \sum_{z_1} \sum_{i=1}^{k} W_{z_1} \gamma_i(z_1) x_i \tag{2.4}$$

where

$$\sum_{z_1} \gamma_0(z_1) = ... = \sum_{z_1} \gamma_k(z_1) = 0$$

or re-written in the style of equation 2.3 as

$$E(Y) = \sum_{z_1} W_{z_1} \left((\beta_0 + \gamma_0(z_1)) + \sum_{i=1}^{k} (\beta_i + \gamma_i(z_1))\, x_i \right) + \sum_{i<j}^{k} \beta_{ij} x_i x_j + \sum_{i=1}^{k} \beta_{ii} x_i^2 \tag{2.5}$$

TABLE 2.1. First-order Model From Draper and John 1988

1	x_1	x_2	z_1	x_1x_2	z_1x_1	z_1x_2
1	-1	-1	-1	1	1	1
1	-1	1	-1	-1	1	-1
1	1	-1	-1	-1	-1	1
1	1	1	-1	1	-1	-1
1	0	0	1	0	0	0

In selecting the best design for fitting the models, Draper and John (1988) uses a more ad hoc approach of listing potential designs according to the initial design conditions and verifying what can be estimated. For a first order departure from the means model, the expectation can be written as $E(y) = \beta_0 + \beta_{0z_1}z_1 + \beta_1x_1 + \beta_2x_2$. (Note, with only two levels of z_1, at -1 and +1, it is possible to express the model instead as $\beta + \beta_{0z_1}z_1 = \beta_0(z_1)$ where $\beta_0 - \beta_{0z_1} = \beta_0(1)$ and $\beta + \beta_{0z_1} = \beta_0(-1)$.) With two quantitative factors and one qualitative factor they suggest five designs in which we partition the four factorial design points between the levels of z_1 and add a center point to one of the levels of z_1. This allows for the estimation of β_1, β_2, β_0 and $\beta_{0z_1}z_1$. What distinguishes these designs is the fifth degree of freedom we get when adding the center point, which allows for the estimation of an extra term in the model. For example, for the design in Table 2.1, the only additional term that can clearly be estimated is x_1x_2 whereas for the design in Table 2.2, we can see that all three possible extra terms are linearly independent and could be estimated if we were interested in them.

To fit the second-order model, they suggest using the four design points plus the center run at each level of z_1 and then partitioning the four axial points amongst the two designs. This keeps with the principle of being able to estimate a full first

TABLE 2.2. First-order Model From Draper and John 1998 b

1	x_1	x_2	z_1	x_1x_2	z_1x_1	z_1x_2
1	-1	-1	-1	1	1	1
1	-1	1	1	-1	-1	1
1	1	-1	-1	-1	-1	1
1	1	1	-1	1	-1	-1
1	0	0	1	0	0	0

order model at all levels of z_1 and the full second order model when we collapse over z_1. They select the best design amongst these by choosing the one with the most options for clearly estimable effects that allows the estimation of all desired effects.

Lastly, John and Draper (1988) discuss fitting designs with two qualitative factors and two or three quantitative factors. The same principles apply that we require at least one point at each combination of z_1 and z_2 levels. If our focus is solely on the first-order model, we must have enough points to clearly estimate the full first-order model when collapsing over all levels of the z_i's.

If our focus is solely on the second-order model, then, at each combination of z_1 and z_2, we require enough points to clearly estimate the full first-order model and enough points to clearly estimate the full second-order model when collapsing over all levels of z_1 and z_2. Again, the suggestion is looking at different partitions and seeing which one provides linear independence in the columns of the design matrix for the desired model terms with special focus on the terms that suggest non-parallel planes if that is of interest.

A more systematic approach is adopted by Wu and Ding (1998) who build on the initial framework and principles of John and Draper (1988). Their focus is on selecting second-order models according to four main criteria:

- The design chosen has enough points to estimate all quantitative second-order terms that do not depend on the z_i's, can estimate the main effects involving the quantitative and qualitative factors, and can estimate all first-order interactions between the qualitative and quantitative factors.
- At each combination of the qualitative factors we have enough points to estimate a full first-order model in the quantitative factors.
- The design is viewed as a sequential design whereby we first estimate the main effects of the qualitative and quantitative factors and the remaining terms are added as a sequential addition to create the second-order model.
- When the design is collapsed over all levels of z_i's, we have enough points to estimate a full second-order model for the quantitative factors.

To create the designs with one qualitative factor, Wu and Ding (1998) begin with an initial 2^{k-p} design that is of high resolution (they do not consider any design of resolution smaller than V). Two center runs and $2k$ star points are added where each factor has one run at $\pm\alpha$ with all other terms held at zero. They take $\alpha = (2^{k-p})^{1/4}$ which satisfies rotatability if the resolution is at least V.

Next, they assign each run to a level of z with the method depending on the number of runs. Regardless of run size, the two center runs are each assigned to a unique level of z_1. If there are 16 runs or fewer in the design, they conduct a systematic computer search over all possible assignments of z_1. If there are more than 16 runs, levels of z_1 for initial design are found by computer search looking at all possible combinations of aliasing the k factors in the factorial points and all possible assignments of z_1 in the additional runs. For designs that are extremely large, where the computer search could be prohibitively long to run, they advocate taking an initial starting design and then either interchanging z_1 assignments to see which improve the optimality criteria (such as in Harville (1974), Wu (1981) and Cook and Nachtsheim (1981)) or switching the signs of individual rows (such as in Wu (1981)).

To select the optimal design, Wu and Ding (1998) calculate and observe various D-optimality criteria that help us ensure our model meets our various design objectives. The seven main criteria considered are:

1. D: Considers the entire design and calculates the traditional D-optimality criterion $|X'X|^{1/n}$ where n is the number of parameters in the model.

2. d_1: Considers the model $(1, x_1, x_2, x_1x_2)$ taking only the runs where $z = 1$.

3. $d_{\text{-}1}$: As above taking only the runs where $z = -1$.

4. $\tilde{D}_1$: Considers the model to be run sequentially in two blocks as $X\beta + ub$ where the block with the original design has its center point assigned to $z = 1$ and the additional design has its center run assigned to $z_1 = -1$. Here b contains the block effect and u is a vector of 1's and -1's depending on the block. The $\tilde{D}_1$ is calculated as a D_s-criterion using the formula $\tilde{D}_1 = |X'X - (X'u)(u'u)^{-1}(u'X)|^{1/n}$.

5. $\tilde{D}_{\text{-}1}$: As above where the original design has its center point assigned to $z_1 = -1$ and the additional design has its center point assigned to $z_1 = 1$.

6. $D_1(i)$: Considers the model $(1, x_1, x_2, z_1, i)$ (where i is either x_1x_2, x_1z_1 or x_2z_1 using only the initial design plus the center point at $z_1 = 1$.

7. $D_{\text{-}1}(i)$: As above only using the center point at $z_1 = -1$.

Criterion 1 emphasizes the need to have an efficient full design. Criteria 2 and 3 emphasize the need to have an efficient design at each level of z_1. Criteria 4 and 5 emphasize the need to have an efficient overall design when it is run in a sequential order. Criteria 6 and 7 emphasize the need to have a good first-order design with whatever additional effect we would like to estimate from the df provided by the inclusion of the center point.

In general, you would like to find designs that have a high D value and approximately equal d_1 and d_{-1} values if both levels of z_1 are of equal interest. Other criteria can be considered depending on our goals.

For designs with two categorical factors, Wu and Ding (1998) suggest selecting an efficient design according to the technique for one categorical factor and, without loss of generality, assigning these runs to $z_2 = 1$. Then the same design is duplicated by reversing the signs for z_1 and switching the value of z_2 to -1. Again this continues with the idea that we should start out with a good initial design and then add on the categorical factors to partition the design after the fact. Obviously there are limitations to this approach as it would involve doubling the size you would need for one categorical factor. This leaves very little option in design size which could be prohibitively expensive. The technique would be nearly completely impractical to extend to three categorical factors as it would involve quadrupling the size of an initial design which would rarely be feasible in practice.

Other developments in RSM designs with qualitative factors, such as Chantarat et al (2003) and Allen and Tseng (2011) focus on a move towards computer generated designs where we are not restricted to factorial and axial points. These papers use the integrated mean square error (IMSE) or expected integrated mean squares error (EIMSE) criteria for finding optimal second-order designs with qualitative factors. These criteria are considered variance + bias criteria as they are concerned with model misspecification. The goal is to not only reduce variance but to reduce bias under misspecification.

In the IMSE criterion, we have a design $\mathbf{D}$ of run size n with k quantitative and r qualitative factors that can be expressed as

$$\underline{\mathbf{y}} = \mathbf{X}_1\boldsymbol{\beta}_1 + \mathbf{X}_2\boldsymbol{\beta}_2 + \boldsymbol{\epsilon} \tag{2.6}$$

where $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ are k_1 and k_2-dimensional vectors corresponding to coefficients that are of primary and secondary interest to us respectively. $\mathbf{X}_1$ and $\mathbf{X}_2$ are the corresponding design matrices and $\boldsymbol{\epsilon}$ is an n-dimensional vector of uncorrelated errors with variance σ^2. Let $\mathbf{f}_1(\mathbf{x})$ and $\mathbf{f}_2(\mathbf{x})$ be the rows in $\mathbf{X}_1$ and $\mathbf{X}_2$ corresponding to the point $\mathbf{x} = \{x_1, \ldots, x_r, z_1, \ldots, z_r\}$. Then we hope to predict

$$\hat{y} = \mathbf{f}_1(\mathbf{x})\hat{\beta}_1 \tag{2.7}$$

where $\hat{\beta}_1 = (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{y}$ assuming that we will fit only the model terms of primary interest after experimentation. Equation 2.7 assumes that the our model is properly specified and can be predicted with ordinary least squares. The IMSE criterion now determines how much bias is introduced if the model is misspecified and $\beta_2 \neq \mathbf{0}$ and is defined as

$$\mathbf{J} = \frac{n}{\sigma^2}\Omega \int_R \mathbf{E}\left[\hat{y}(\mathbf{x}) - \eta(\mathbf{x})\right]^2 d\mathbf{x} \tag{2.8}$$

where R is our design space, Ω is the reciprocal of its volume and $\eta(\mathbf{x}) = \mathbf{X}_1\beta_1 + \mathbf{X}_2\beta_2$. It can be shown that equation 2.8 is equivalent to variance + bias.

In practice however, the IMSE criterion is generally of minimal use as it requires an assumption that we know the value of β_2. As an alternative, the more practical criterion is the EIMSE criterion which assumes that β_2 is a random variable and can be simulated using Monte Carlo estimation where we are only required to make assumptions on its variance structure. The EIMSE criterion is given by:

$$\begin{aligned} EIMSE(\mathbf{D}) &= E_{\mathbf{x},\beta_1,\beta_2,\epsilon}\left\{\left[\hat{y}(\mathbf{x},\mathbf{D},\beta_1,\epsilon) - \eta(\mathbf{x},\beta_1,\beta_2)\right]^2\right\} && (2.9)\\ &= \sigma^2 Tr\left[\mu_1 1(\mathbf{X}_1'\mathbf{X}_1)^{-1}\right] + Tr\left[\mathbf{K}_2\Delta\right] && (2.10)\end{aligned}$$

where

$$\begin{aligned}
\mathbf{K_2} &= E\left[\beta_2\beta_2'\right] \\
\Delta &= \mathbf{A}'\mu_{11}\mathbf{A} - \mu_{12}'\mathbf{A} - \mathbf{A}'\mu_{12} + \mu_{22} \\
\mathbf{A} &= (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2 \\
\mu_{ij} &= \Omega\int_R \mathbf{f}_j(\mathbf{x})'\mathbf{f}_i(\mathbf{x})d\mathbf{x} \text{ for } i,j = 1,2 \text{ and } i \leq j
\end{aligned}$$

Thus $\mathbf{K}_2$ is our assumed covariance structure for β_2 and the μ_{ij} are our moment matrices. To calculate the criterion we apply a Bayesian setting by assuming a distribution for β_2 and giving diffuse priors to the β_1 terms.

To deal with the qualitative factors in optimization, the problem is treated as an integer programming problem over a candidate set of points $\mathbf{C}$, where $\mathbf{C}$ is a $(k+r)\times N$ matrix. If $\mathbf{C}_j$ is the j^{th} row in the candidate set, the integer programming programming is equivalent to:

$$\begin{aligned}
\text{Minimize} \quad & \text{EIMSE}(\mathbf{D}), \\
\text{subject to} \quad & \mathbf{D} = \begin{bmatrix} C_{I_1} \\ \vdots \\ C_{I_n} \end{bmatrix} \text{ where } \{I_1, ..., I_n\} \subset \{1, ..., N\}
\end{aligned}$$

The candidate set of points considered by Allen and Tseng (2011) were either:

- EIMSE I: Equally spaced points $-\alpha, -\alpha/2, 0\alpha/2, \alpha$ in the quantitative factors
- EIMSE II: Non-equally spaced points $-\alpha, -1, 0, 1, \alpha$ in the quantitative factors

- EIMSE III: Uniformly distributed random samples in the design region.

where α is the radius of R and the qualitative factors take on their pre-determined values.

For various values of k and r, Allen and Tseng (2011) compared their optimal EIMSE designs under all three types of candidate sets and compared them to the designs recommended in John and Draper (1988) and Wu and Ding (1998) and found their designs to work better under most circumstances with respect to model misspecification. Their designs were not as optimal under D-efficiency but had competitive results.

Apart from a loss in D-optimality, another downside to the use of the EIMSE criterion is that it is harder to implement in practice as a statistician, as it is not programmed into readily available software. The authors were able to perform the integer programming using the Custom Design feature in JMP and using the algorithm in Hadj-Alouane and Bean (1997).

Now with an understanding of some approaches to assigning z_1, we will revisit our notion of specifying the appropriate model.

There are two general approaches for setting up the model for categorical factors which we have seen thus far. We could specify our model as a departure from the mean, as in Equation 2.4, or we could be using an indicator model, such as in Equation 2.3, where we have separate coefficients in some of the terms at each level of z_1. Both will lead to the same fitted responses (as we will demonstrate) but they will have their own unique design matrix.

Consider an experiment with two quantitative factors and one categorical factor with two levels. Draper and John (1988) consider the following design:

x_1	x_2	z
-1	-1	-1
-1	1	-1
1	-1	-1
1	1	-1
-1	-1	1
-1	1	1
1	-1	1
1	1	1
$-\sqrt{2}$	0	1
$\sqrt{2}$	0	1
0	$-\sqrt{2}$	-1
0	$\sqrt{2}$	-1
0	0	-1
0	0	1

This has enough points to clearly estimate a full first-order design with a centre point at each level of z and partitions the four axial points amongst the two levels of z.

John and Draper (1998) use this design to fit the following model in the departure from means format,

$$\begin{aligned} E(y) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{z_1} z_1 + \beta_{1z_1} x_1 z_1 + \beta_{2z_1} x_2 z_1 \\ + \beta_{12} x_1 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2. \end{aligned} \tag{2.11}$$

We note that this model has common second-order terms that are not dependent on z_1. This is partly due to an insufficient number of runs to clearly estimate all terms in a full second-order model but further reasoning will be discussed later in the chapter.

The design matrix used in analysis would be as follows,

$$\mathbf{X}_1 = \begin{bmatrix} 1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 & -1 & 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -\sqrt{2} & 0 & 1 & -\sqrt{2} & 0 & 0 & 2 & 0 \\ 1 & \sqrt{2} & 0 & 1 & \sqrt{2} & 0 & 0 & 2 & 0 \\ 1 & 0 & -\sqrt{2} & -1 & 0 & \sqrt{2} & 0 & 0 & 2 \\ 1 & 0 & \sqrt{2} & -1 & 0 & -\sqrt{2} & 0 & 0 & 2 \\ 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

where the columns correspond to I, x_1, x_2, z, x_1z, x_2z, x_1x_2, x_1^2, and x_2^2.

As in standard linear regression, the coefficients are estimated by least-squares as

$$\begin{aligned} \underline{\hat{\beta}} &= (\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_z, \hat{\beta_{1z}}, \hat{\beta_{2z}}, \hat{\beta_{12}}, \hat{\beta_{11}}, \hat{\beta_{22}})' \\ &= (\mathbf{X_1}'\mathbf{X_1})^{-1}\mathbf{X_1}'\underline{y}. \end{aligned} \tag{2.12}$$

To test whether any of our coefficients are nonzero, we form an ANOVA table as follows:

Source	df	Sum of Squares	Mean Sum of Squares	F
Regression	9	SST-SSE	SSR/9	MSR/MSE
Error	5	$\underline{y}'\underline{y} - \underline{\hat{\beta}}'\mathbf{X_1}'\underline{y}$	SSE/5	
Total	14	$\underline{y}'\underline{y}$		

and then test $H_0 : \beta_1 = \beta_2 = \ldots = \beta_{22} = 0$ vs H_A : At least one of the β's does not equal zero.

If we are interested in the significance of individual co-efficients, we would test the hypothesis

$$H_0 : \beta_i = 0 \text{ vs. } H_A : \beta_i \neq 0,$$

using the test statistic

$$t = \frac{\hat{\beta}_i - 0}{\sqrt{MSE * \mathbf{C}[i,i]}}.$$

where $\mathbf{C}[i,i] = (\mathbf{X'X})^{-1}[i,i]$.

If we are interested in testing the significance of a group of coefficients, this would be done by conducting an F-test which compares the error sums of squares from a full and reduced model.

Alternatively, we could be in the second situation and attempting to fit a separate model for the first-order terms at each level of the categorical variable using the indicator model

$$\begin{aligned} E(y) = \quad & W_{-1}\left[\beta_0(-1) + \beta_1(-1)x_1 + \beta_2(-1)x_2\right] + W_{+1}\left[\beta_0(+1) + \beta_1(+1)x_1 + \beta_2(+1)x_2\right] \\ & + \beta_{12}x_1x_2 + \beta_{11}x_1^2 + \beta_{22}x_2^2. \end{aligned} \tag{2.13}$$

To have the separate estimates for each level of z_1, we need to rewrite the design matrix so that each factor which is affected by z_1 has a column that is non-zero strictly where $z_1 n$ is at the appropriate level as below. The design matrix is as follows,

$$
\mathbf{X}_2 = \begin{bmatrix}
1 & 0 & -1 & 0 & -1 & 0 & 1 & 1 & 1 \\
1 & 0 & -1 & 0 & 1 & 0 & -1 & 1 & 1 \\
1 & 0 & 1 & 0 & -1 & 0 & -1 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & -1 & 0 & -1 & 1 & 1 & 1 \\
0 & 1 & 0 & -1 & 0 & 1 & -1 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & -1 & -1 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & -\sqrt{2} & 0 & 0 & 0 & 2 & 0 \\
0 & 1 & 0 & \sqrt{2} & 0 & 0 & 0 & 2 & 0 \\
1 & 0 & 0 & 0 & -\sqrt{2} & 0 & 0 & 0 & 2 \\
1 & 0 & 0 & 0 & \sqrt{2} & 0 & 0 & 0 & 2 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix},
$$

where the columns correspond to $I(-1)$, $I(1)$, $x_1(1)$, $x_1(-1)$, $x_2(1)$, $x_2(-1)$, x_1x_2, x_1^2, and x_2^2.

The estimates, ANOVA calculations and tests on the individual coefficients are conducted the same way as in situation 1.

Again, both models will agree in the analysis and we can find the coefficients and variance estimates for each of the regression coefficients from the analysis of the other model.

To demonstrate, note that we can relabel our coefficients for ease of analysis as follows,

$$\begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_z \\ \beta_{1z} \\ \beta_{2z} \\ \beta_{12} \\ \beta_{11} \\ \beta_{22} \end{pmatrix} = \begin{pmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \\ \delta_5 \\ \delta_6 \\ \delta_7 \\ \delta_8 \\ \delta_9 \end{pmatrix} \qquad \begin{pmatrix} \beta_0(-1) \\ \beta_0(+1) \\ \beta_1(-1) \\ \beta_1(+1) \\ \beta_2(-1) \\ \beta_2(+1) \\ \beta_{12} \\ \beta_{11} \\ \beta_{22} \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \\ \alpha_9 \end{pmatrix}.$$

As, for example, we see that $\beta_0(-1) = \beta_0 - \beta_z$ we can relate the vectors of coefficients as follows:

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \\ \alpha_9 \end{pmatrix} = \begin{pmatrix} \delta_1 - \delta_4 \\ \delta_1 + \delta_4 \\ \delta_2 - \delta_5 \\ \delta_2 + \delta_5 \\ \delta_3 - \delta_6 \\ \delta_3 + \delta_6 \\ \delta_7 \\ \delta_8 \\ \delta_9 \end{pmatrix}$$

or

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \\ \alpha_9 \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \\ \delta_5 \\ \delta_6 \\ \delta_7 \\ \delta_8 \\ \delta_9 \end{pmatrix}$$

i.e. $\underline{\boldsymbol{\alpha}} = \mathbf{B}\underline{\boldsymbol{\delta}}$.

To get the variance-covariance of one from the other we can relate them as:

$$\begin{aligned} c\hat{o}v(\hat{\underline{\alpha}}) &= cov(\mathbf{B}\hat{\underline{\delta}}) \\ &= \mathbf{B}cov(\hat{\underline{\delta}})\mathbf{B}' \\ &= \mathbf{B}(\mathbf{X_1}'\mathbf{X_1})^{-1}\mathbf{B}' * MSE. \end{aligned}$$

Despite showing that the two models are in a sense “equivalent”, we choose in this thesis to focus on the second indicator variable method. Firstly as it is consistent with main literature for RSM with categorical factors. However, more importantly, it helps us focus on what we are trying to accomplish. We would like to estimate separate response surfaces for each level of the categorical variable. This not only allows us to compare them directly but it allows us to test coefficients separately or for curvature in the model at both levels of the categorical factor as it may not be the same answer for both. We are treating each level of the categorical factor as if it may have its own unique surface and so we choose to make the model in a way where it is easy to separate the different response surfaces.

In general, when choosing a model, it is also important to keep in mind our goals in RSM as well. Common goals include: modelling the entire response surface, finding the maximum and/or minimum (the extrema) of the response surface at every level of z_1 or finding the maximizing/minimizing value (which we will call the optimizing value) of the response surface at every level of z_1. Depending on our goal, different coefficients may be of more importance to us than others. In the two models we have considered thus far, we have assumed common second-order terms which will largely be for when we focus on finding the optimizing value. If we are concerned with finding the actual extrema, we will need to consider a third model that includes separately estimated coefficients for the second-order terms at each level of z_1.

Though we are not focusing on differences in second-order terms in the thesis, we should look at a simple example to motivate why this may occur and point towards potential future work. Let us examine what happens in the single variable problem. Consider the following second-order model with one quantitative random variable and one categorical random variable in standard form,

$$E(Y) = W_1\left[\beta_0(1) + \beta_1(1)x_1 + \beta_2(1)x_1^2\right] + W_2\left[\beta_0(2) + \beta_1(2)x_1 + \beta_2(2)x_1^2\right] \quad (2.14)$$

Another way to write the model is in vertex form

$$E(Y) = W_1\left[\gamma_0(1) + \gamma_2(1)(x_1 - \gamma_1(1))^2\right] + W_2\left[\gamma_0(2) + \gamma_2(2)(x_1 - \gamma_1(2))^2\right], \quad (2.15)$$

which emphasizes more meaningful parameters in terms of vertical and horizontal shifts and stretches.

Algebraically we can find that,

$$\gamma_0(i) = \beta_0(i) - \frac{\beta_1(i)^2}{4\beta_2(i)} \tag{2.16}$$

$$\gamma_1(i) = \frac{\beta_1(i)}{2\beta_2(i)} \tag{2.17}$$

$$\gamma_2(i) = \beta_2(i). \tag{2.18}$$

The term $\gamma_1(i)$ represents the horizontal shift of the parabola which is also equivalent to the optimizing value. The term $\gamma_0(i)$ is the vertical shift of the parabola and is also equivalent to the extrema of the response surface (i.e. the height of the response surface at the optimizing value). The term $\gamma_2(i)$ represents the vertical stretch factor of the parabola. This term tends to be of less interest to us and often assumed not to depend on z.

The γ terms all have a meaningful interpretation, which is not the case for the β terms. However, we do not generally use equation 2.15 for fitting the data as it is a nonlinear model.

We have established that the two sets of coefficients are connected and it is possible to go back and forth between them. Thus, we can estimate the β's first and then estimate the γ's from the β's using approximate standard errors based on Tyler-series type appoximations.

When our data involves categorical factors, an important point of investigation in RSM will often be if each level of the categorical factor has the same extrema or the same optimizing value. In what follows we will see that this motivates why we usually use a model that uses a common second-order term at each level of z.

As $\beta_2(i) = \gamma_2(i)$, the β_2 term is equivalent to the vertical stretch term. If γ_2 or β_2 is equal at each level of z_1, then testing whether the optimizing value is the same for each level of z_1 (whether $\gamma_1(1) = \gamma_1(2)$), becomes equivalent to testing whether $\frac{\beta_1(1)}{4\beta_2} = \frac{\beta_1(2)}{4\beta_2}$. This holds when $\beta_1(1) = \beta_1(2)$ which is now a linear test instead of a non-linear test due to the assumption that the second-order term is the same at each level of z.

Note, this is not true for testing whether $\gamma_0(1) = \gamma_0(2)$ due to the presence of the square term in the $\beta_1(i)$ and so it will remain a non-linear test even with the simplifying assumption.

This helps to motivate why in the previous models we had common second-order terms. While it could be forced due to a lack of runs to estimate the second-order terms, it may be a conscious decision to make the simplifying assumption for the ease of analysis when our goal in RSM is to find the optimizing value and the exact shape of the response curve is not of interest.

Now we will consider the two variable problem. This will allow us to examine how our coefficients control different translations, rotations and stretches of our response surface. Knowledge of this will allow us to investigate which coefficients will be important for our different goals of RSM.

Considering a model with just two quantitative random variables, there are two different ways that we could write our model for the second-order model. We can write it out as a linear regression model as:

$$E(Y) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \beta_{12} x_1 x_2 \tag{2.19}$$

or in the vertex form of:

$$\begin{aligned} E(Y) = &\frac{((x_1 - \alpha_1)\cos(\alpha_5) + (x_2 - \alpha_2)\sin(\alpha_5))^2}{\alpha_3^2} \\ &+ \frac{((x_1 - \alpha_1)\sin(\alpha_5) - (x_2\alpha_2)\cos(\alpha_5))^2}{\alpha_4^2} + \alpha_6. \end{aligned} \tag{2.20}$$

In the vertex form of model 2.20, there are six parameters that affect the shape of the response surface at the cross section for each level of z_1.

The coefficient α_5 is a rotation parameter that rotates the axis of the conic section by α_5 degrees counter-clockwise as pictured in Figure 2.1.

The coefficients α_1 and α_2 give the location of the center of the conic section relative to the rotated axis as in Figure 2.3.

The coefficients α_3 and α_4 gives the radius of the conic section on the rotated axis for the x_1 and x_2 axis respectively as in Figure 2.2.

The parameter α_6 affects the height of the response surface though, unlike the one-variable problem, it does not give the value of the minimum/maximum. It merely affects its location by shifting the 3-D surface vertically. It also comes in to play by affecting the width at each fixed level of y.

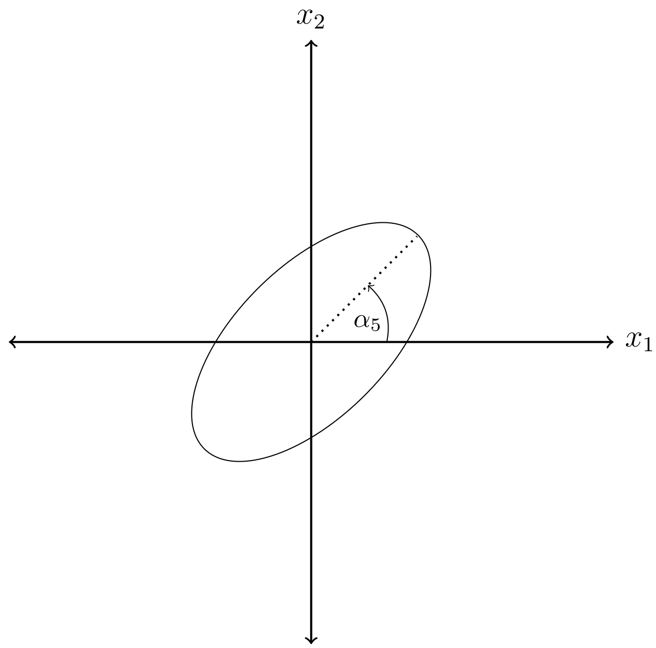

FIGURE 2.1. Rotated Standard Ellipse

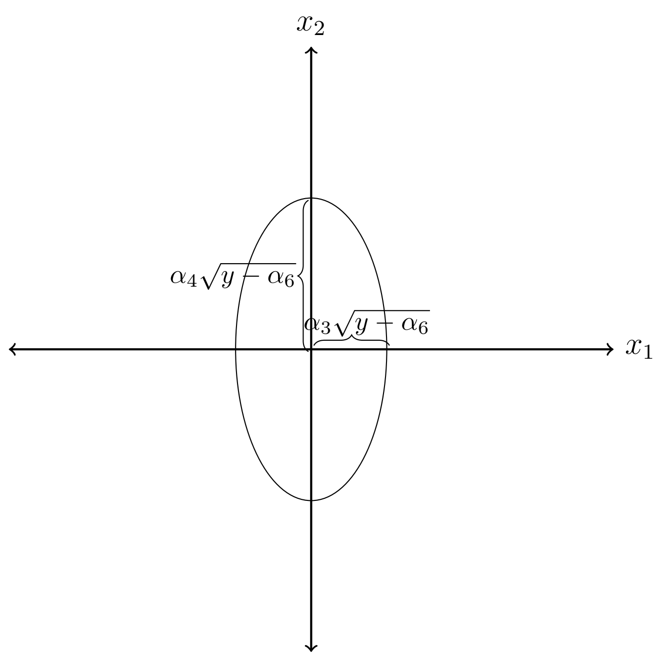

FIGURE 2.2. Standard Ellipse

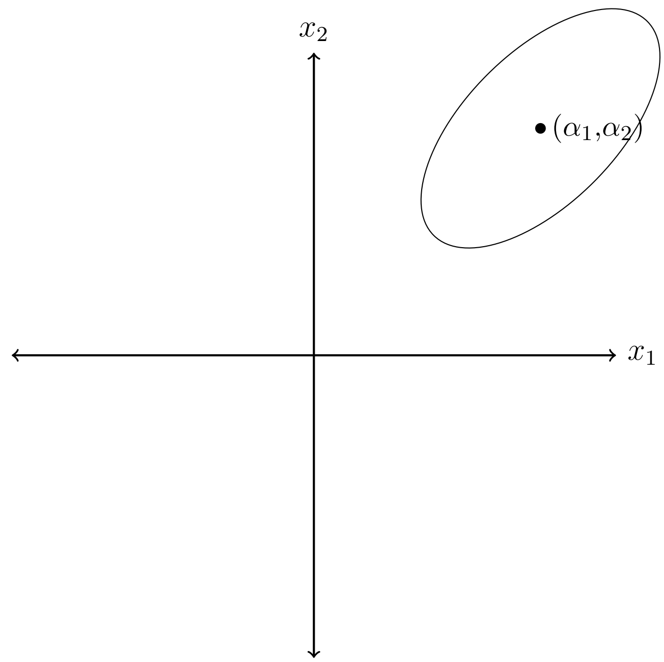

FIGURE 2.3. Rotated and Shifted Standard Ellipse

To see the connections between them we can perform algebra on model 2.20 to get the relationships between then β's and the α's as follows:

$$
\begin{aligned}
\beta_{11} &= \frac{\cos^2(\alpha_5)}{\alpha_3^2} + \frac{\sin^2(\alpha_5)}{\alpha_4^2} && (2.21) \\
\beta_{22} &= \frac{\sin^2(\alpha_5)}{\alpha_3^2} + \frac{\cos^2(\alpha_5)}{\alpha_4^2} && (2.22) \\
\beta_{12} &= \frac{\sin(2\alpha_5)}{\alpha_3^2} - \frac{\sin(2\alpha_5)}{\alpha_4^2} && (2.23) \\
\beta_{1} &= -\frac{2\alpha_1\cos^2(\alpha_5) + \alpha_2\sin(2\alpha_5)}{\alpha_3^2} - \frac{2\alpha_1\sin^2(\alpha_5) - \alpha_2\sin(2\alpha_5)}{\alpha_4^2} && (2.24) \\
\beta_{2} &= -\frac{2\alpha_2\sin^2(\alpha_5) + \alpha_1\sin(2\alpha_5)}{\alpha_3^2} - \frac{2\alpha_2\cos^2(\alpha_5) - \alpha_1\sin(2\alpha_5)}{\alpha_4^2} && (2.25) \\
\beta_{0} &= \frac{(\alpha_2\sin(\alpha_5) + \alpha_1\cos(\alpha_5))^2}{\alpha_3^2} + \frac{(\alpha_2\cos(\alpha_5) - \alpha_1\sin(\alpha_5))^2}{\alpha_4^2} + \alpha_6. && (2.26)
\end{aligned}
$$

If there is no rotation and, hence, $\alpha_5 = 0$, the terms become easily invertible and we can derive the α's in terms of the β's as:

$$
\begin{aligned}
\alpha_1 &= \frac{-\beta_1}{2\beta_{11}} && (2.27) \\
\alpha_2 &= \frac{-\beta_2}{2\beta_{22}} && (2.28) \\
\alpha_3 &= \frac{1}{\sqrt{\beta_{11}}} && (2.29) \\
\alpha_4 &= \frac{1}{\sqrt{\beta_{22}}} && (2.30) \\
\alpha_5 &= \beta_0 - \frac{\beta_1^2}{4\beta_{11}^2} - \frac{\beta_2^2}{4\beta_{22}}. && (2.31)
\end{aligned}
$$

Again, we can see the advantage of being able to assume that the second order terms of the regression model are the same for both levels of z_1. This will allow us to test that the locations parameters are the same as a linear test by comparing only the $\beta_1(i)$'s and $\beta_2(i)$'s.

While we have dealt with the shape and location of the response surface, we also have our third goal under consideration for RSM of finding the optimizing value for the response surface. To do this we take the partial derivative of model 2.19 with respect to both x_1 and x_2 to get:

$$\frac{\partial y}{\partial x_1} = \beta_1 + 2\beta_{11}x_1 + \beta_{12}x_2 \tag{2.32}$$

$$\frac{\partial y}{\partial x_2} = \beta_2 + 2\beta_{22}x_2 + \beta_{12}x_1. \tag{2.33}$$

By setting the partial derivatives to zero and solving for x_1 and x_2 we can find the optimizing values to be:

$$\begin{aligned}\begin{pmatrix} x_1^* \\ x_2^* \end{pmatrix} &= \frac{-1}{4\beta_{11}\beta_{22} - \beta_{12}^2}\begin{bmatrix} 2\beta_{22} & -\beta_{12} \\ -\beta_{12} & 2\beta_{22} \end{bmatrix}\begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} \\ &= \begin{pmatrix} (2\beta_{22}\beta_1 - \beta_{12}\beta_2)/(4\beta_{11}\beta_{22} - \beta_{12}^2) \\ (-\beta_{12}\beta_1 + 2\beta_{11}\beta_2)(4\beta_{11}\beta_{22} - \beta_{12}^2) \end{pmatrix}. \end{aligned} \tag{2.34}$$

From equation 2.34, we can see that the optimizing value is a function of the second-order terms multiplied by the vector of the first-order terms excluding the intercept. Again, if we are able assume that the second-order terms are the same for both levels of z_1 then testing for equivalence of the optimizing value at both levels of z_1 becomes equivalent to testing whether the $\beta_1(i)$'s and $\beta_2(i)$'s are equal which is a linear test.

Having a linear departure from means model, a linear model with indicator functions, or a non-linear model in point-intercept form can create a big difference in what we are able to estimate and what information we may garner. While they are all "equivalent" in the sense that they should lead to the same estimates upon re-arranging and combining terms, the coefficients play different roles. We chose this particular model for our goal of separate response surfaces and having a linear test for practitioners but there is potential work using others as well.

We can now see the importance of specifying our model and have an understanding of how to choose an appropriate design for the chosen model when we have a categorical factor in RSM. Next we will look at the current literature for a different type of response surface design, the split-plot design.

2.2 Split-Plot Designs

In chapter 1, we introduced a special type of design common in agricultural and industrial experiments, the split-plot design. Split-plot designs were first introduced by Fisher (1925) with applications in agriculture. They have found further applications in industrial experimentation and elsewhere.

As we learned in chapter one, split-plot designs arise when some factors are harder to vary than others. This could be because of physical, time, or cost constraints.

These harder-to-vary factors are called whole-plot factors and the easy-to-vary factors are called split-plot factors.

In Jones and Nachtsheim (2009), the authors give an agricultural experiment as an example which demonstrates the origin of the split-plot name. In their example, there are four fields of land available for use. The two factors are irrigation method (factor A) and fertilizer (factor B), both of which have two different levels.

However, there is a complication. While fertilizer can easily be changed for smaller pieces of land, it is not physically possible for an irrigation method to be used on any sized land less than an entire field. Thus, if we were applying our factors, each entire field would need to receive the same irrigation method but we could change the fertilizer within the field. The hard-to-vary factor needs to be randomized less "often". To conduct the experiment, each field is assigned at random either irrigation method 1 or 2. Then each field is split in half. At random, one half of each field will get fertilizer A and the other will get fertilizer B. The whole-plot factor remains constant for the entire field. The field is then literally split in half, with each half receiving one level of the split-plot factor.

A logical way to conduct this experiment, though it is not mentioned in Jones and Nachsteim (2009), is to conduct the experiment with replication in two blocks. Visually, we can see this in Figure 2.4. Within each block (here the block is a grouping on two fields that are near in proximity), we randomly assign each field to $A = 1$ or $A = -1$. Within each of the fields, we assign the sub-plot to either $B = -1$ and $B = +1$. The design can be found in Table 2.3.

The important thing to note about split-plot designs, is they are different from other factorial designs in the way we conduct the run order. At first glance, split-plot

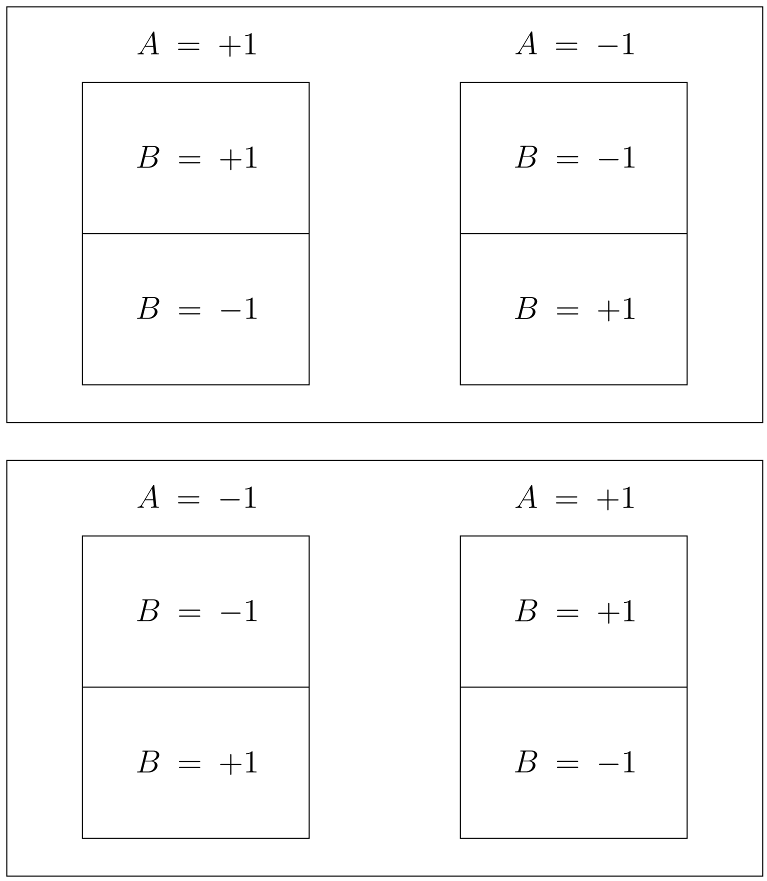

FIGURE 2.4. A split-plot agricultural experiment with one two-level whole-plot factor A and one two-level split-plot factor B replicated in two blocks.

Block	A	B
-1	-1	1
-1		-1
-1	1	1
-1		-1
1	-1	1
1		-1
1	1	1
1		-1

TABLE 2.3. A split-plot design with one whole-plot factor and one split-plot factor in an agricultural experiment.

designs may look to be the same as a completely randomized design, but the run order will vary which makes them more complicated to analyze. It also follows that some factors and effects are estimated with less precision than others.

Consider Table 2.4. It shows the columns for potential designs with three factors: A, B, C. There is also a potential blocking variable that will be used for most designs. This could be a variable such as time if the experiment is performed over several days. In an agricultural experiment we could be using farmland in different regions of the province for the same experiment.

One design that could be described using factors A, B, and C is a 2^3 full factorial design with replication. This is seen in the column labeled "CRD". All possible $2^3 = 8$ run combinations at two-levels are used and appear twice for a total of 16 runs. Note that all 16 runs are done completely at random with no restrictions on the order in which they are run.

Our possible effects and available degrees of freedom are shown in table 2.5. We see that we have enough degrees of freedom to estimate all effects and also have enough for estimating the error variance.

Block	A	B	C	CRD	RCBD	1 WP / 2 SP	2 WP / 1 SP
-1	-1	-1	-1	12	3	2	1
-1	-1	-1	1	9	4	4	2
-1	-1	1	-1	10	6	1	6
-1	-1	1	1	3	5	3	5
-1	1	-1	-1	1	2	8	7
-1	1	-1	1	13	1	7	8
-1	1	1	-1	15	7	5	3
-1	1	1	1	7	8	6	4
1	-1	-1	-1	8	10	13	11
1	-1	-1	1	11	9	14	12
1	-1	1	-1	4	13	15	16
1	-1	1	1	6	16	16	15
1	1	-1	-1	14	12	12	14
1	1	-1	1	2	15	9	13
1	1	1	-1	5	11	10	9
1	1	1	1	16	14	11	10

TABLE 2.4. Run order for various 2^3 designs with replication run with and without a split-plot structure.

Effect	df
A	1
B	1
C	1
AB	1
AC	1
BC	1
ABC	1
Error	8
Total	15

TABLE 2.5. Effects and associated degrees of freedom for a full CRD 2^3 design with two replicates.

Another possibility is to run a 2^3 full factorial design with replication, but with replication in blocks. This is called a randomized complete block design (RCBD). That is to say, the runs are grouped together by a variable that cannot be randomized, such as day, if we need to run half the experiment on day one and the other half on day two. Day would be the blocking variable. Another example is an industrial experiment that could be blocked by the technician operating the machinery or an agriculture experiment that could be blocked by putting fields that are close together into the same block. We note that these blocking variables are variables that may be affecting our experiment but are not able to be meaningfully replicated outside of the experiment. While the operator may affect our experiment, when the process is done in practice, it will not necessarily be the same technician operating the process.

This design uses factors A, B, C, and the blocking variable. The run order is shown in the "RCBD" column of table 2.4. All eight possible treatment combinations are run once in a completely random order in the first block and then they are all run again in a completely random order in the second block.

If we assume that there is no interaction between the blocks and the factors, then we will get the effects and associated degrees of freedom in table 2.6. This is a reasonable assumption. Moreover, without this assumption, the analysis would usually be more complex. This lack of treatment$\times$block interaction is important. If blocks interacted with the treatments, then all 7 error degrees of freedom would be used to estimate those treatment$\times$block interaction effects leaving no degrees of freedom for error. Our hope is that the blocking variable merely has an additive effect, because it is of little interest by itself, and it cannot be replicated outside of the experimental setting.

Effect	df
Blocks	1
A	1
B	1
C	1
AB	1
AC	1
BC	1
ABC	1
Error	7
Total	15

TABLE 2.6. Effects and associated degrees of freedom for a full 2^3 design replicated in two blocks.

The next design we consider is a split-plot design with one whole-plot factor (A) and two split-plot factors (B, C) with replication run in two blocks. Full factorial split-plot designs are typically denoted as $2^{(n_1+n_2)}$ designs; thus, this would be a $2^{(1+2)}$ design, replicated in two blocks. The run order is shown in column "1 WP / 2 SP" of Table 2.4. In the first block, we see that the whole-plot factor A is constant for four runs at a time. We select at random whether to do the first four entries of the design or the second four entries first. Then, within those four entries (called a whole plot), we randomly select the run order of the four runs. This is then repeated for the other whole plot. In this particular example, the first whole plot is chosen first with the rows run in the order (2, 4, 1, 3) and then the second whole plot is run with the rows run in order (8, 7, 5, 6). This is repeated once again in the second block. First we select at random which whole plot to use (here we run the second whole plot) and then within each of those whole plots, we have randomized the run order.

In table 2.7, we can see how the degrees of freedom are assigned. As there are two randomization processes (selecting which whole plots to run and then selecting the run-order within the whole plots), there are two error terms and each of our main effects and interaction terms is associated with one of those error terms depending on how often it varies.

There are three terms that only change when the whole plot changes and never within a whole plot: blocks, A, and blocks$\times A$. In order to be able to estimate the whole-plot error term, we need assume that one of these terms is unimportant. With four whole plots, there are only 4-1=3 degrees of freedom available for the whole plot plus error terms. Therefore, we will generally assume that it is the whole plot$\times$ blocks interaction effect that is unimportant as with the RCBD and put that variation in to the whole-plot error term.

If we exclude block interaction effects, there are 6 main effects and interaction effects that vary within whole plots: B, C, AB, AC, BC, and ABC. The remaining degrees of freedom are generally attributed to the split-plot error term. They are essentially the blocks$\times$factor interaction effects which we have deemed to be unimportant.

Next we consider the split-plot design with two whole-plot factors (A, B) and one split-plot factor C run with replication in two blocks, a $2^{(2+1)}$ design. The run order is shown in column "2 WP / 1 SP" of Table 2.4. Here we have whole plots of size two as our whole-plot factors A and B remain constant for two runs at a time creating four whole-plots within each block. We run the experiment once in block 1 and then a second time in block 2. Within each block, we first pick a run order at random for the four whole plots. In this particular example we ran them in order 1,

Effect	df
Blocks	1
A	1
Whole-plot Error	1
B	1
C	1
AB	1
AC	1
BC	1
ABC	1
Subplot Error	6
Total	15

TABLE 2.7. Effects and associated degrees of freedom for a 2^{1+2} split-plot design replicated in two blocks.

3, 4, and 2, respectively. Within each of the whole plots we randomly choose the run-order of the runs in that whole plot. So within the first whole plot we first do run 1 and then run 2. In the second whole plot we would run the second run first (denoted 6, 5 as it is actually the third whole plot to be run though it is the second whole plot in the design). This is then repeated for the second block.

In Table 2.8 we can see the associated degrees of freedom for this design. As there are 8 whole-plots, there are 8-1=7 degrees of freedom for the effects randomized at the whole-plot level, plus the whole-plot error term. Effects blocks, A, B, and AB all vary strictly at the whole-plot level and will each have one degree of freedom. If the three effect×block interaction terms are assumed unimportant they are used to estimate the whole-plot error.

This leaves 16-7-1=8 degrees of freedom remaining for the split-plot effects and associated error term. As C, AC, BC, and ABC all vary at the split-plot level, they each receive one degree of freedom. The remaining 4 degrees of freedom are given to

Effect	df
Blocks	1
A	1
B	1
AB	1
Whole-plot Error	3
C	1
AC	1
BC	1
ABC	1
Subplot Error	4
Total	15

TABLE 2.8. Effects and associated degrees of freedom for a 2^{2+1} split-plot design replicated in two blocks.

estimating the split-plot error term.

We notice the common theme in all of these designs, the replication is necessary because we need to use the seven block×effect interaction terms to estimate our error terms. In a completely randomized design without replication (i.e. running only half of the A, B, C columns with no run-order restrictions), we have 7 effects that need estimating and only 8-1=7 degrees of freedom. This means we have nothing left to estimate the error term. In such a situation, we would want to see if it is at all possible to add extra runs. If it is not, a normal probability plot could be used to determine which factor effects are likely unimportant and combine them into the error term. Now let us see how this works in the split-plot setting.

Suppose we were running the last design with two whole-plot and one split-plot factors but we did not have the second block worth of runs and there was no replication. We would be forced to distribute the 8-1=7 degrees of freedom as in table 2.9.

Effect	df
A	1
B	1
Whole-plot Error (AB)	1
C	1
AC	1
BC	1
ABC	1
Subplot Error	Not Estimable
Total	7

TABLE 2.9. Effects and associated degrees of freedom for a 2^{1+2} split-plot design without replication.

There are four whole plots so 4-1=3 degrees of freedom associated with whole plots. As A and B are main effects, we are forced to use AB as the estimate of whole-plot error. There are four other main effects and interaction terms that vary at the split-plot level: C, AC, BC, and ABC. As there are only four remaining degrees of freedom, in order to be able to estimate all of our effects we would not have any degrees of freedom for the sub-plot error. Thus, we would need to decide which of these effects are unimportant to combine them into an error term. In any event, we need to make judicious choice in how we estimate our error term and without replication, we may need to sacrifice two-factor interactions in order to do so.

The point is, even though designs may look the same on the surface, we need to pay careful attention to how they are run, as this plays an important role in the analysis. Split-plot designs in particular need to be recognized due to their unique structure with two different error terms.

We can summarize, in general, the usual model for a split-plot design and its

ANOVA table. Consider a split-plot experiment involving one whole-pot factor, A, with a levels, and one split-plot factor, B, with b levels where the design is randomized as a randomized complete block design (RCBD) with c blocks. The model is given in Montgomery (2009) as:

$$y_{ijk} = \mu + \alpha_i + \gamma_k + \alpha\gamma_{ik} + \beta_j + \alpha\beta_{ij} + \epsilon_{ijk}, \tag{2.35}$$

for $i = 1, 2, \ldots, a; j = 1, 2, \ldots, b; k = 1, 2, \ldots, c$; where $\sum \alpha_i = 0$, $\sum \beta_j = 0$, $\sum_i \alpha\beta_{ij} = 0 \ \forall j$, $\sum_j \alpha\beta_{ij} = 0 \ \forall i$, $\gamma_k : IN(0, \sigma^2_\gamma)$, $\alpha\gamma_{ik} : IN(0, \sigma^2_{\alpha\gamma})$, $\epsilon_{ijk} : IN(0, \sigma^2)$, and all of the γ_k's, $\alpha\gamma_{ik}$'s and ϵ_{ijk}'s are independent.

The ANOVA table for this model is given by

Source	DF	Expected Mean Square
A	$a-1$	$\sigma^2 + b\sigma^2_{\alpha\gamma} + \frac{bc}{a-1}\sum \alpha_i^2$
Block	$c-1$	$\sigma^2 + b\sigma^2_{\alpha\gamma} + ab\sigma^2_\gamma$
WP-error	$(a-1)(c-1)$	$\sigma^2 + b\sigma^2_{\alpha\gamma}$
B	$b-1$	$\sigma^2 + \frac{ac}{b-1}\sum \beta_j^2$
AB	$(a-1)(b-1)$	$\sigma^2 + \frac{c}{(a-1)(b-1)}\sum\sum \alpha\beta_{ij}^2$
SP-error	$a(b-1)(c-1)$	σ^2
Total	$abc-1$	

We can see that this is structured under the assumption that the blocks$\times$ factor interaction terms are not significant. The split-plot designs above, all conformed to this structure.

So far, the four designs we have looked at were all full factorial designs where we had at least one replication of each factor combination. With only three factors and

Run	x_1	x_2	$x_3 = x_1x_2$
1	-1	-1	1
2	-1	1	-1
3	1	-1	1
4	1	1	-1

TABLE 2.10. A 2^{3-1} fractional factorial design.

no center points or axial points for a second-order design, the number of required runs is quite small. However, industrial experiments can have many factors and require many more runs in addition to the factorial points if we are estimating a second-order design. If a full factorial design leads to too many runs for practical purposes, an option is to run a fractional factorial design. In a fractional factorial design where we are restricted to 2^{n-k} runs, we can run every combination of the first $n-k$ factors and then generate the remaining k added factors by multiplying some combination of the non-generated factors.

For example, for a design with 2^{3-1} runs, we would take every possible combination of the first two factors x_1 and x_2 and then generate x_3 by multiplying $x_1 \times x_2$ so that $x_3 = x_1x_2$ as seen in Table 2.10. Note, this is not a very practical design due to the small number of runs but is used for illustration purposes only.

When we generate factors it introduces an important relationship into the structure of the design. Namely, if $x_3 = x_1x_2$ then we must have that $I = x_1x_2x_3$ where I is the identity vector $(1, 1, 1, 1)'$. This relationship conforms to the properties of a group where each element is its own multiplicative inverse. The group structure also implies that $x_1 = x_2x_3$ and $x_2 = x_1x_3$. These "chains" of effects cannot be distinguished from one another in analysis and we will not be able to tell which factor effect or interaction is driving the response. By having half as many runs as a full factorial design, we will have half of our effects aliased with the other half in pairs. Hence, we would generally like to generate our added factors by multiplying as many columns together as possible so that our main effects will not be aliased with other lower-order effects. If we were, for example, to see in another design that $x_1 = x_2x_3x_4x_5$, we could likely safely make the assumption that the effect is due to the main effect x_1 and not the four-way interaction $x_2x_3x_4x_5$.

In a split-plot setting, we stated that full factorial designs are denoted as $2^{n_1+n_2}$ where n_1 is the number of whole-plot factors and n_2 is the number of split-plot factors. A fractional factorial split-plot design that has k_1 generated whole-plot factors and k_2 generated split-plot factors is denoted by $2^{(n_1+n_2)-(k_1+k_2)}$. An example of a $2^{(2+2)-(1+1)}$ design with three whole-plot factors and three split-plot factors is seen in Table 2.11. The whole plots are of size $2^{3-1} = 4$ and we generated X_3 by X_1X_2 and x_3 by $X_1x_1x_2$.

When we are generating factors in a rational factorial split-plot design there are certain restrictions as given in Bingham and Sitter (1999):

1. Whole-plot factors can only be generated by other whole-plot factors.
2. Split-plot factors must be generated using at least one other split-plot factor.

These restrictions will ensure that added whole-plot factors will not vary more often than the original whole-plot factors and that added split-plot factors will continue to vary at the sub-plot level. Despite the generation, the randomization

Run	X_1	X_2	$X_3 = X_1X_2$	x_1	x_2	$x_3 = X_1x_1x_2$
1	-1	-1	1	-1	-1	-1
2				-1	1	1
3				1	-1	1
4				1	1	-1
5	-1	1	-1	-1	-1	1
6				-1	1	-1
7				1	-1	1
8				1	1	-1
9	1	-1	-1	-1	-1	1
10				-1	1	-1
11				1	-1	-1
12				1	1	1
13	1	1	1	-1	-1	1
14				-1	1	-1
15				1	-1	-1
16				1	1	1

TABLE 2.11. A $2^{(3+3)-(1+1)}$ fractional factorial split-plot design with generators $X_3 = X_1X_2$ and $x_2 = X_1x_1x_2$.

rules remain the same as before. We will potentially just have fewer whole plots or less runs per whole plot than in the full factorial setting.

On a final note for the structure and analysis of first-order split-plot designs, we should be mindful that fractional factorial designs may need to make extensive use of normal probability plots to select the appropriate factors for inclusion in the chosen fitted model. As the generation will create many chains of effects that are aliased, we will only be able to estimate one effect from each such chain. Thus, if $x_3 = x_1x_2$, there is ambiguity over which effect is driving the response and what we are estimating is likely a combination of their effects.

Another issue with aliasing, is that it may effect whether a term is a split-plot effect (and tested against the split-plot error term) or a whole-plot effect (and tested against the whole-plot error term). In full factorial split-plot designs, an effect is a whole-plot effect if it contains only whole-plot factors and split-plot effect is an effect that contains at least one split-plot factor. However, in a fractional factorial split-plot design, due to aliasing, we may be surprised to find that an effect that would normally be considered a split-plot effect is aliased with a whole-plot effect and would thus be tested against the whole-plot error.

When we test for significant effects, there will be two normal probability plots to examine, one for split-plot effects and one for whole-plot effects. It is important that we are mindful of which factors are included in which plot when selecting our likely significant effects.

2.2.1 Second-order Split-Plot Designs

With an understanding of first-order full factorial and fractional factorial split-plot designs, we will now move on to second-order designs. Generally, though not always, second-order split-plot designs are made by taking a full factorial or fractional factorial design and then adding center points and axial points to help estimate the terms in a second-order model. We will begin by looking at the model for second-order split-plot designs and then consider several common design types used in the literature to estimate the model terms.

For a second-order split-plot design, Vining, Kowalski, and Montgomery (2005) give the model with m whole-plot factors and k split-plot factors as

$$y_{ij} = f_w(\mathbf{X}_i)'\beta_{wp} + \delta_i + f_s(\mathbf{X}_i, \mathbf{x}_j)'\theta_{sp} + \epsilon_{ij} \tag{2.36}$$

where

- $f_w(\mathbf{X}_i)'\beta_{wp} = \beta_0 + \beta_1 X_1 + ... + \beta_m X_m + \beta_{12}X_1X_2 + ... + \beta_{(m-1)m}X_{m-1}X_m + \beta_{11}X_1^2 + ... + \beta_{mm}X_m^2$, the model terms for the whole-plot effects,
- $f_s(\mathbf{X}_i, \mathbf{x}_j)'\theta_{sp} = \theta_0 + \theta_1 x_1 + ... + \theta_k x_k + \theta_{12}x_1x_2 + ... + \theta_{(k-1)k}x_{k-1}x_k + \theta_{11}x_1^2 + ... + \theta_{kk}x_k^2 + \theta_{k+1}X_1x_1 + ... + \theta_{k+mk}X_mx_k$, the model terms for the split-plot effects,
- $\delta_i \sim N(0, \sigma_\delta^2)$,
- $\epsilon_{ij} \sim N(0, \sigma^2)$,

σ_δ^2 is the whole-plot error variance and σ^2 is the sub-plot error variance. Again, these two error terms are an important feature of split-plot designs. By Vining, Kowalski, and Montgomery (2005), the variance components can be estimated to test the model terms as long as there is sufficient replication.

In matrix form, Vining, Kowalski, and Montgomery give the model with N runs and p coefficients as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\delta} + \boldsymbol{\epsilon} \tag{2.37}$$

where

- $\mathbf{y}$ is the $N \times 1$ response vectors
- $\mathbf{X}$ is the $N \times p$ design matrix
- $\boldsymbol{\beta}$ is the vector of coefficients
- $\boldsymbol{\delta}$ is an $N \times 1$ vectors of random whole-plot error terms
- $\boldsymbol{\epsilon}$ is an $N \times 1$ vector of random sub-plot error terms

The variance-covariance matrix for y is given by

$$\boldsymbol{\Sigma} = \sigma^2\mathbf{I} + \sigma_\delta^2\mathbf{J} \tag{2.38}$$

where

$$\mathbf{J} = \begin{bmatrix} \mathbf{1}_1\mathbf{1}_1' & 0 & \cdots & 0 \\ 0 & \mathbf{1}_2\mathbf{1}_2' & \cdots & 0 \\ & & & \\ 0 & 0 & \cdots & \mathbf{1}_m\mathbf{1}_m' \end{bmatrix}$$

and $\mathbf{1}_i$ is a vector of 1's equal to the numbers of runs within whole-plot i for $i = 1, 2, ..., m$.

The GLS estimate of $\boldsymbol{\beta}$ is given by

$$\hat{\beta} = (\mathbf{X}'\mathbf{\Sigma}^{-1}\mathbf{X})^{-1}\mathbf{X}'\mathbf{\Sigma}^{-1}\mathbf{y} \tag{2.39}$$

with variance covariance matrix

$$cov(\hat{\beta}) = (\mathbf{X}'\mathbf{\Sigma}^{-1}\mathbf{X})^{-1}. \tag{2.40}$$

Depending on the chosen design, the model is often estimated by using restricted maximum likelihood techniques or by establishing an equivalence of ordinary least-squares estimators with the generalized least-squares estimators where it is appropriate. Knowing the proposed model for a second-order split-plot design, we can now look at various choices of designs that can be used to estimate this model.

Throughout the past three decades there have been a wide variety of approaches to selecting designs for second-order split-plot designs that began with focuses on star and cube designs in Draper and John (1998). More recent years have had some focus on two-strata rotatability (Wang, Vining, Kowalski 2010), and a much larger focus on various forms of equivalent estimation designs in papers such as Wang, Kowalski, and Vining (2009); Parker, Kowalski, and Vining (2007a), Parker, Kowalski, and Vining (2007b); and Vining and Kowalski (2008). We will consider the construction of some of these designs as they will be used as our base designs for creating RSM split-plot designs that include categorical factors.

2.2.2 Star and Cube Designs

In Draper and John (1998), the authors advocate the use of star and cube designs. These designs are run on the premise that it may be impractical to have 5 levels of a hard-to-vary factor and, as such, they are limited to having just three levels: -1, 0, 1.

The cube design is run much like a central composite design with the elimination of the axial points for the hard-to-vary factors. The design has four components. First, there are runs where the split-plot factors individually vary from $\pm\alpha$ while the whole-plot factors are 0. This is followed by runs where the whole-plot factors run a full-factorial design while, at the subplot level, the split-plot factors vary from $\pm b$.

For example, the cube design for 2 whole-plot and 2 split-plot factors is given in Figure 2.12.

For the STAR designs we have one part of the design where the hard-to-vary factor remains 0 while the easy-to-vary factor varies simultaneously from $\pm a$, a second part where the the hard-to-vary factor simultaneously varies from ± 1 while the easy-to-vary factor varies individually from $\pm b$, plus n_c center points.

For a design with two whole-plot factors and two split-plot factors the design is given in Figure 2.13.

In Draper and John (1998), the authors give tables of values of a and b for various combinations of numbers of whole-plot and split-plot factors that maximize a rotatability measure denoted by Q^*. A rotatable design, in a non-split-plot response surface context, would have an equal amount of precision in the estimation of all of the factors. This is measured by the prediction variance in terms of distance to the

X_1	X_2	x_1	x_2
-1	-1	$-b$	$-b$
		$-b$	b
		b	$-b$
		b	b
-1	1	$-b$	$-b$
		$-b$	b
		b	$-b$
		b	b
1	-1	$-b$	$-b$
		$-b$	b
		b	$-b$
		b	b
1	1	$-b$	$-b$
		$-b$	b
		b	$-b$
		b	b
0	0	$-a$	0
		a	0
		0	$-a$
		0	a
0	0	0	0
		0	0
		0	0
		0	0

TABLE 2.12. A CUBE design for two hard-to-vary and two easy-to-vary factors.

X_1	X_2	x_1	x_2
0	0	$-a$	$-a$
		$-a$	a
		a	$-a$
		a	$-a$
-1	-1	$-b$	0
		b	0
		0	$-b$
		0	b
-1	1	$-b$	0
		b	0
		0	$-b$
		0	b
1	-1	$-b$	0
		b	0
		0	$-b$
		0	b
1	1	$-b$	0
		b	0
		0	$-b$
		0	b
0	0	0	0
		0	0
		0	0
		0	0

TABLE 2.13. A STAR design with two hard-to-vary and two easy-to-vary factors.

center. Draper and John (1998) acknowledge that this is not possible in a split-plot structure but the authors strive to make it as rotatable as possible. Draper and John (1998) suggest as well that Box-Behnken designs may be a possibility for one difficult to change factor but with more whole-plot factors, Box-Behnken designs lead to too many whole plots.

2.2.3 Two-Strata Rotatable Designs

In Wang (2006) and Wang, Vining, and Kowalski (2010) the authors expand on the idea of rotatability in split-plot designs. They prove that we can not achieve true rotatability in a split-plot design, and they advocate for a design that is two strata-rotatable.

In a two-strata rotatable design, distance to the centre is measured separately for the whole-plot factors and for the split-plot factors. That is, the whole-plot factors are rotatable when the split-plot factors are held constant and the split-plot factors are rotatable when the whole-plot factors are held constant. Wang, Vining, and Kowalski (2010) defined this explicitly as follows:

"In a k-factor split-plot design with k_1 WP factors and k_2 SP factors, let z_i be the ith WP factor and x_j be the jth SP factor. If for any two design runs $t_0 = (z_{10}, z_{20}, .., z_{k1,0}, x_{10}, ..., x_{k2,0})$ and $t_1 = (z_{11}, z_{21}, ..., z_{k1,1}, x_{11}, ..., x_{k2,1})$ satisfying $\Sigma_{i=1}^{i=k_i} z_{i1}^2 = \Sigma_{i=1}^{i=k_1} z_{i1}^2 = \rho_w^2$ and $\Sigma_{j=1}^{j=k_2} x_{j0}^2 = \Sigma_{j=1}^{j=k_2} x_{j1}^2 = \rho_s^2$, the corresponding prediction variances $Var(\hat{y}(t_0))$ and $Var(\hat{y}(t_1))$ are equal, the design is called a two-strata rotatable design."

In the paper, the authors give values of α and β (Wang, Vining, and Kowalski's notation for a and b) for several unbalanced and balanced designs, in particular the VKM balanced and unbalanced CCDs. They make a point of separating designs by whether or not the second-order split-plot terms vary at the whole-plot level as they must be treated separately.

They also note that the addition of additional center runs does not affect the rotatability of the design. This is important to note as, later in the thesis, for the purposes of having equal sized whole-plots, we add center runs to some designs to achieve an equal number of runs for the center run whole-plot.

2.2.4 Equivalent Estimation Designs

A large focus in split-plot RSM has been on the creation of equivalent estimation designs. OLS-GLS equivalent designs are treated in papers such as Vining, Kowalski, and Montgomery (2005); Goos (2006), Parker (2007); Parker, Kowalski, and Vining (2007a); Parker, Kowalski, Vining (2007b), Macharia and Goos (2010), Jones and Goos (2012a) and Nguyen and Pham (2015).

This class of designs is named because the model estimates are the same using both generalized least-squares (GLS) and ordinary least-squares (OLS). The main advantages to being able to use OLS is it leads to an amount of "comfort" for practitioners as they can now use exact tests with known degrees of freedom with inference for any sample size (Vining and Kowalski, 2008).

Vining, Kowalski, and Montgomery (2005) establish these general conditions for OLS-GLS equivalence:

1. the design is balanced, i.e. each whole plot contains the same number of subplots;

2. the subplot designs are orthogonal (not necessarily the same design);

3. the axial runs for the subplot factors are run in a single whole plot.

In Parker (2007), he expands on equivalent estimation designs by allowing for unequal sized whole plots. These allow for the center run whole plots to have less runs than other whole plots while still maintaining OLS-GLS equivalence.

Parker, Kowalski, and Vining (2007a) continue to advocate for OLS-GLS equivalent designs and list many more advantages to the designs: they require no assumption on the variances ratios, the equivalence property can hold for projections, parameters are BLUE and are independent of variance components. They suggest using designs with a minimum number of whole plots and using a natural number of sub-plots per whole plot if one is available.

The Parker, Kowalski, and Vining (2007a) designs use a construction method based on taking a CCD and re-arranging the points so that the hard-to-vary factors remain constant within the whole plots and the split-plot factors are orthogonal. The axial points of the WP factors go into whole plots by themselves, all center points are put into whole plots containing only center points, and the split-plot factors are replicated as much as necessary to maintain balance in these additional whole plots. Their appendix list several of their designs.

In Parker, Kowalski, and Vining (2007b), the authors examine unbalanced designs. They suggest that balance may not be desirable if split-plot factors are costly to reset or if we are requiring extensive replication of split-plot centre points solely to achieve balance. Their paper includes an example where an experiment is run under extreme environmental conditions. It is costly to maintain whole plots for too long of time so minimizing sub-plots per whole plot is at a premium. Under these circumstances, these may be the appropriate designs to use.

2.2.5 Minimum Whole Plot Designs

An extension of equivalent estimation designs come in the form of minimum whole plot designs. Parker, Kowalski, and Vining (2007a) give a method for constructing split-plot central composite designs that have a minimum number of whole plots. They define minimum whole plot designs as designs where "the number of WPs, m, is equal to the number of runs in a completely randomized design based solely on the whole-plot factors".

The paper establishes the minimum number of whole plots that are required for a split-plot central composite design. Of particular note in the minimum whole plot designs is that the whole plot sizes are odd and there are SP center points within each whole plot. This is necessary as they establish that we cannot have a whole plot containing all 0's in both the whole-plot and split-plot factors and maintain equivalence. The equivalence property for their minimum whole plot designs is model dependent and assumes that the complete second-order model will be kept. These designs can be found in the Parker equivalent estimation catalog. For all other designs, examples can be found in the appendix.

2.2.6 Definitive Screening Designs

Cheng and Wu (2001) suggested an alternative approach to the typical two-stage experimental process of first conducting a screening experiment to eliminate unnecessary effects and then secondly conducting a response surface design to fit the final model with a reduced number of effects. The authors proposed a one-stage process that would both screen for important effects and be useful for fitting the final response surface using a three-level design without having a tremendous number of runs (which is why this is traditionally done in two steps). Designs are chosen based on the number of projections with fewer factors included and with the most D-efficient projections.

Building on this notion, Jones and Nachtcheim (2011b) proposed a new class of designs called definitive screening designs (DSDs). These designs are desirable due to their run size (double the number of factors plus 1), orthogonal main effects, and main effects that are independent of second-order terms. In Jones and Nachtsheim (2013), they extend definitive screening designs to include two-level categorical factors. Other treatments of definitive screening experiments can be found in Xiao et al. (2012), Georgiou et al. (2014), and Lin (2015).

The authors in Lin and Yang (2015) extended definitive screening designs to the split-plot setting. They studied the designs looking at eligible projections as well as their D-efficiency and I-efficiency. The authors concluded that the definitive screening designs performed well when there were relatively few factors of importance.

It is likely these designs will continue to receive considerable attention in the literature but they will not be considered in the current catalog in the thesis due to their unbalanced structure and often having whole plots of size one, which will not lend themselves to our methods for adding categorical factors. It is likely they will need specific consideration in the future if we would like to incorporate categorical factors into RSM split-plot DSDs.

In the next chapter, we will take the CUBE designs, two-strata rotatable designs, and equivalent estimation designs found in the Appendix and use them to create split-plot designs with categorical factors. These designs, found in the literature, will be used as base designs as we describe methods for appending categorical factors. This will be followed up with catalogs of the optimal assignments for each base design according to D-optimality.

Chapter 3

RSM Split-plot Designs with Categorical Factors

3.1 Introduction to RSM Split-plot Designs with Categorical Factors

In Chapters 1 and 2, we explored ways in which categorical factors were added to RSM designs with quantitative factors, with emphasis on the works of Draper and John. We recognized that, it was entirely possible that response surfaces could be different at each level of the categorical factor and in many different ways: curves could be shifted vertically or horizontally, they could have a stretching factor applied to them, or they may be completely unrelated. Each of these modifications were important for addressing our main questions in RSM: What is the optimum? What is the optimizing value? How does the response surface look? The answer to these questions could be different for each separate instance of the categorical factor but, this is difficult to address if we do not separate out the different response surfaces.

We discussed in Chapter 2 how, with separate response surfaces for each level of the categorical factor, we could derive tests for equality of the optimizing values.

Categorical factors can come up often in both industrial and agricultural settings. In industrial experimentation, it could be due to using different chemical components or a different operating technique. In agriculture, we could be using different brands of fertilizer or using a different planting method. Not only is it normal for these experiments to have categorical factors, but they also lend themselves well to split-plot designs.

In the previous chapters, we explored how the split-plot design has a unique structure and has special needs and considerations when it comes to analysis. The RSM split-plot designs with quantitative factors are inherently different in nature than standard central composite designs in the non-split-plot setting. It is a natural extension to ask how to properly introduce categorical factors for response surface methodology in split-plot designs.

At a cursory glance, we can see some obvious challenges up front. Firstly, there are shared challenges with non-split-plot RSM such as factors needing more than just two levels. Secondly, there are challenges that apply to the split-plot design in particular, the randomization restrictions. Previously, in the method of Draper and John (1988), the authors were using a search algorithm that chose the levels of the categorical factor through a random search across the entire design space with no heed to whether or not consecutive rows in the design had the same value of the categorical factor. This is not something that can be done in a split-plot design. For whole-plot factors, the factor levels must remain consistent for a number of runs at a time. For the split-plot factors, we have concerns about balance within the sub-plots. All of this will play an important factor in how we decide to add the categorical factor.

However, before we begin to add the categorical factors, we will introduce some notation that explains how we will describe split-plot designs with various structures. To distinguish between response surface designs containing categorical factors, we will adopt the notation of $RS\,(i, j; k, l)$ where:

- i is the number of WP quantitative factors,
- j is the number of WP categorical factors,
- k is the number of SP quantitative factors,
- l is the number of SP categorical factors.

Note that if $i = j = 0$ or $k = l = 0$ then we have no WP factors or no SP factors, respectively, and the design is not a split-plot design.

Recall that we are also using the notation:

- X_i's are quantitative WP factors
- x_i's are quantitative SP factors
- Z_i's are categorical WP factors
- z_i's are categorical SP factors.

Using this notation and taking into account the unique structure, this chapter will focus on important concerns for how categorical factors in split-plot designs should be added, explanations of how to practically add categorical factors addressing these concerns, and a detailing of an R program to find D-optimal split-plot designs under various settings. The chapter will culminate with a table of D-optimal designs.

3.2 General Principles

Before looking at how we will assign the categorical factor in depth, we will first recall the conditions required in the non-split-plot setting with Draper and John (1988) and Wu and Ding (1998).

Firstly, we know that we required to be able to estimate a full first-order model in the quantitative variables at each level of the categorical factor. Practically speaking for us, this means that we require the subsetted designs (the two designs induced when separating out the rows in the original design by the level of the categorical variable) to be capable of clearly estimating the terms in the first-order model. This can be achieved by ensuring that the factorial points are partitioned in a way that allows the subsetted matrices to each be orthogonal. Our assignment method will need to take this into account.

Secondly, we must have a center point at each level of the categorical factor so that we can estimate curvature for a response surface for each level of the categorical factor. This will require attention, particularly if all the center points are contained in the same whole plot. If this is true and we are adding a whole-plot categorical factor, then we will need to make sure that additional center points are added to the design.

Thirdly, we must be able to estimate a full second-order design when we collapse over the categorical variable. As we are picking designs in the literature that are already being advocated as good by the authors, we will already know they are appropriate designs for this purpose.

These basic principles will guide our selection of the design with the categorical factor. We will advocate starting with a good RSM split-plot design that has both quantitative split-plot and whole-plot factors. This design will be orthogonal as a whole, as well as in the partitioned designs for the factorial points so that all first-order terms will be orthogonal at each level of the categorical variable and in the design as a whole. As required in Draper and John (1988) and Wu and Ding (1998), this ensures that we can clearly estimate all first-order terms in the design as a whole and for the separate response surfaces at each level of the categorical factor. Adhering to these principles, we will be able create appropriate designs that will create estimable response surfaces for each of the categorical factors while maintaining a split-plot structure in the subsetted designs.

In the next section, we will look at four different approaches that could be used to assign the levels of the categorical factor that satisfy the conditions we have set out. We will follow that with instructions on how to find the D-optimal design for each of the four methods for a given base design (the originally chosen quantitative RSM split-plot design). Finally we will make tables of D-optimal designs for the various methods using popular RSM split-plot designs in the literature and show an example of using the algorithm to add a categorical factor to a split-plot experiment.

3.3 Approaches to Adding a Categorical Factor

With the main principles in mind, there are four main approaches that could be taken for assigning the levels of the categorical factor to a split-plot design. For a given design, the method used will be dependent upon whether we are adding a split-plot or a whole-plot factor.

To demonstrate the main ideas behind each of these methods, we will look at some simple full factorial split-plot designs to demonstrate the basic concepts and then follow this up with examples in RSM designs. We will follow this with a discussion of how the algorithm carries out three of these methods and explore through example the limitations of each method.

3.3.1 Method 1: Adding a Whole-plot Categorical Factor By Doubling the Design

The simplest approach that we can take to creating a categorical RSM split-plot design is to simply "double" the design. We will take a good base design in our quantitative factors and repeat each whole plot. For one instance of each whole plot, we will set the categorical factor to $+1$ and for the other instance of the whole plot, we will set the categorical factor to -1.

Consider the 8-run full factorial split-plot design in Table 3.1 with two quantitative whole-plot factors and one quantitative split-plot factor. It has 4 whole plots, each with two runs.

If we add a whole-plot categorical factor by doubling the design, we would get the design in Table 3.2. Note that we are not blocking by Z_1. To run this design,

X_1	X_2	x_1
-1	-1	-1
		1
-1	1	-1
		1
1	-1	-1
		1
1	1	-1
		1

TABLE 3.1. A 2^{2+1} full factorial split-plot design.

we still pick one of the eight whole plots completely at random and then randomly select the run order within the whole plot. This is repeated until all whole plots have been run.

This approach would give you the most information and be the easiest to design from a theoretical standpoint. When we extend this to a second-order design, it would also ensure that at each level of Z_1 we would have a "good" second-order design (our base design was already considered a good design). However, except for very small designs such as here, this method is generally not very practical.

While it would be ideal theoretically, doubling the size of an experiment could be costly and we presumably picked a good design to begin with that was in the realm of the number of runs that were desirable. We work under the premise in this thesis that time and runs are at a premium and we will make our decisions accordingly.

Z_1	X_1	X_2	x_1
-1	-1	-1	-1
			1
-1	-1	1	-1
			1
-1	1	-1	-1
			1
-1	1	1	-1
			1
1	-1	-1	-1
			1
1	-1	1	-1
			1
1	1	-1	-1
			1
1	1	1	-1
			1

TABLE 3.2. Adding a whole-plot factor by doubling the design

3.3.2 Adding a Categorical Whole-Plot Factor: Pseudo-Partitioning

In Chapter 1 we saw how Draper and John (1988) advocated a straightforward partitioning of a first-order design for assigning categorical factors. We could attempt, in theory, to apply this approach to our second-order designs for adding a categorical whole-plot factor.

In a split-plot context, partitioning would mean partitioning not individual runs to the various levels of the categorical factor but partitioning entire whole plots to have level -1 or 1 for all the runs within the whole plot. This would maintain the split-plot structure and not reduce the number of whole plots.

For the factorial points, the method would work as such. Reconsider the full factorial design in Table 3.1. If we partition the four whole plots then two whole plots would get $Z_1 = 1$ and two whole plots would get $Z_1 = -1$ as in Table 3.3. While this works nicely for the factorial points, we will see that this method falls apart when we consider a second-order design.

Consider the design in Table A.1. We notice, for example, that all of the center points belong to the same whole plot. As well, all of the axial points for the split-plot factors belong to the same whole plot. If we were to perform a straightforward partitioning of the points, then we would not have a "good" design at each level of the categorical factor. Only one level would receive center points and we would only be able to estimate second-order terms for the split-plot factors at one level of the categorical factor.

This is a common trend in second-order split-plot designs to put all of the axial

Z_1	X_1	X_2	x_1
-1	-1	-1	-1
			1
1	-1	1	-1
			1
1	1	-1	-1
			1
-1	1	1	-1
			1

TABLE 3.3. 2^{2+1} full factorial split-plot design with whole plots partitioned

points for any particular factor in the same whole plot. Sometimes the center points are distributed throughout several whole plots, such as in minimum whole plot designs, but most often they are not.

Clearly this is an issue that makes straightforward partitioning not a viable option. It is clear that we need enough points to have a reasonable model at each level of the categorical factor and partitioning does not often achieve this.

A compromise to this would be what I call pseudo-partitioning. This is a hybrid of doubling the design and partitioning the whole plots. For the factorial whole plots, each whole plot will receive either $Z_1 = 1$ or $Z_1 = -1$, a strict partitioning as shown in Table 3.3. For the added whole plots, we will create a second copy of each such that one instance of each whole plot receives $Z_1 = 1$ and $Z_1 = -1$ (as in doubling the design).

We note that this method will increase both the number of runs and the number of whole plots required. Increasing the number of whole plots may be an issue if the

X_1	x_1	x_2
-1	-1	-1
	-1	1
	1	-1
	1	1
1	-1	-1
	-1	1
	1	-1
	1	1

TABLE 3.4. A 2^{1+2} full factorial split-plot design.

WP factors are extremely difficult to change. Adding runs may also be undesirable but we must have some sort of trade off.

3.3.3 Adding a Categorical Whole-Plot Factor: Splitting the Whole Plots

The third approach for adding a whole-plot categorical factor involves splitting the whole plots. Assuming that there are more than 3 runs per whole plot, if we absolutely cannot increase the number of runs due to time or expense, we can consider adding an additional whole plot factor by re-assigning half the runs in each existing whole plot to $Z_1 = 1$ and half to $Z_1 = -1$.

Let us look at how this would be done for the factorial points using the full factorial 2^{1+2} design in Table 3.4 with one quantitative whole-plot factor and two quantitative split-plot factors.

Z_1	X_1	x_1	x_2		Z_1	X_1	x_1	x_2
-1	-1	-1	-1		-1	-1	-1	-1
1		-1	1				1	1
1		1	-1		1	-1	-1	1
-1		1	1	$\Longrightarrow$			1	-1
1	1	-1	-1		-1	1	-1	1
-1		-1	1				1	-1
-1		1	-1		1	1	1	1
1		1	1				-1	-1

TABLE 3.5. Adding a categorical WP by splitting whole plots

There are two whole plots with four runs each. When we add Z_1, we will randomly assign within each of these whole plots two runs where $Z_1 = 1$ and two runs where $Z_1 = -1$. If needed, we will then re-order the design so that it is arranged properly into whole plots. We can see one such assignment and re-arrangement in Table 3.5.

We now have an 8-run design with four whole plots and two sub-plots per whole plot. The number of runs has been held intact but we now have twice as many whole plots. While this is merely a demonstration of the method, splitting the whole plots for axial points works in the same way.

This method is the only one for adding whole plots that does not increase the number of runs but it does add the most whole plots.

The method you choose will depend on the importance of your concerns of number of whole plots vs number of runs, how difficult the levels are to change, and restrictions on which designs can be used for different methods, which we will explore later in the chapter.

X_1	X_2	x_1	z_1
-1	-1	-1	-1
		1	1
-1	1	-1	1
		1	-1
1	-1	-1	1
		1	-1
1	1	-1	-1
		1	1

TABLE 3.6. 2^{2+2} with an added split-plot factor.

3.3.4 Adding a Split-Plot Factor

Adding a split-plot factor is the most straight forward process with one option. As there are no randomization restrictions on the split-plot factors, we can randomly assign half of the runs to receives $z_1 = -1$ and half of the runs to receive $z_1 = 1$.

However, if we require balance in our design, we will need to ensure that we are randomly assigning the runs within the whole plots individually and not the entire design. Thus, instead of assigning half of the runs in the entire design to $z_1 = 1$ or $z_1 = -1$, we will do so for each whole plot separately.

Reconsider the full 2^{1+2} design in Table 3.4. It has two whole plots with four sub-plots per whole plot. Within each of the whole plots, we will assign two runs to get $z_1 = 1$ and hence the other two runs to get $z_1 = -1$.

Note that the same method will apply to adding the categorical factor for the axial points.

$$\mathbf{X}^{-} = \begin{array}{ccc} X_1 & X_2 & x_1 \\ \hline -1 & -1 & -1 \\ -1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \end{array} \quad \mathbf{X}^{+} = \begin{array}{ccc} X_1 & X_2 & x_1 \\ \hline -1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & -1 \\ 1 & 1 & 1 \end{array}$$

TABLE 3.7. Subsetted designs $\mathbf{X}^{-}$ and $\mathbf{X}^{+}$ respectively where all columns are orthogonal.

3.3.5 Further Considerations

For each of our methods, we must ensure that we are meeting the basic criterion that we are able to clearly estimate the coefficients for the first-order model at each level of the categorical variable. Thus, it will be important in our algorithm to make sure that we have enough points to clearly estimate the first-order model at each of the levels of the categorical variable. This will need to be taken in to account when we search for optimal designs to ensure that any design being inspected is feasible.

The way this can be observed is by partitioning the factorial points of the design matrix $\mathbf{X}$ into two sub-designs: $\mathbf{X}^{-}$ and $\mathbf{X}^{+}$. Here $\mathbf{X}^{-}$ contains all of the runs in the factorial points where the categorical variable equals -1 and $\mathbf{X}^{+}$ contains all of the runs in the factorial points where the categorical variables equals $+1$. These designs are known collectively as the subsetted designs.

Let us consider the design in Figure 3.6 where we added a categorical split-plot factor as factor z_1 in the righthand column. If we split it into the subsetted designs where $z_1 = 1$ and $z_1 = -1$ we get the matrices in Table 3.7

In both of the subsetted designs we can see that all columns are orthogonal to

X_1	x_1	x_2	z_1
-1	-1	-1	1
	-1	1	-1
	1	-1	-1
	1	1	1
1	-1	-1	1
	-1	1	-1
	1	-1	-1
	1	1	1

TABLE 3.8. A 2^{1+2} design in the quantitative factors with an added split-plot categorical factor.

one another. This means we will have a clear estimate of all the first-order factor effects.

Now, consider instead the assignment of z_1 to the design in Table 3.8 which had a 2^{1+2} base design.

When we take the subsetted designs we get the matrices in Table 3.9. We can see that in $\mathbf{X}^+$, $x_1x_2 = -\mathbf{I}$, implying $x_1 = -x_2$. In $\mathbf{X}^-$, we have $x_1x_2 = \mathbf{I}$, implying $x_1 = x_2$. Thus we cannot estimate both the effects of x_1 and x_2 at either level of the categorical variable. As such we cannot clearly estimate a full first-order design at both levels of z_1 and the assignment is not eligible.

As a consequence, after making any possible assignment of a categorical variable we will need to check not only that the full design matrix is orthogonal but that the subsetted designs in the factorial points are orthogonal as well. This will ensure we have a feasible assignment.

$$\mathbf{X}^{-} = \begin{array}{ccc} X_1 & x_1 & x_2 \\ \hline -1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \end{array} \quad \mathbf{X}^{+} = \begin{array}{ccc} X_1 & x_1 & x_2 \\ \hline -1 & -1 & -1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ 1 & 1 & 1 \end{array}$$

TABLE 3.9. Subsetted designs $\mathbf{X}^{-}$ and $\mathbf{X}^{+}$ respectively where all columns are not orthogonal.

After assigning axial points, we will also check that the entire design matrix continues to be orthogonal.

3.4 Optimal Designs and the Search Algorithm

Now that we have a clear understanding of the four different approaches to adding categorical factors, we can explore our algorithm. At this point we assume that the practitioner has decided to use an $RS\,(i, j; k, l)$ design and has selected a good split-plot base design with i quantitative whole-plot factors and k quantitative split-plot factors. They now need to pick a method and a D-optimal allocation of the categorical factor which we will find using an R program.

In this section, I will outline what information the practitioner needs for input, what is meant by an "optimal design" relative to the program, how the algorithm works for each method, what the program prints out, and how to interpret the output. I will also discuss the computational and processing time challenges and what has been done to address them. As well, I will discuss the limitations of the program in its current form and give suggestions for future additions.

To use the program, first an $RS\,(i,0;j,0)$ base design needs to be selected according to what the practitioner feels is a good split-plot design. Again, this is to allow for the case where the categorical factor has no effect and the design should be collapsed over the categorical factor. Several such RSM split-plot designs were introduced in Chapter 2.

First the base design needs to be converted into a design matrix in R. The following information needs to be collected initially for input into the algorithm:

- nwp: Number of whole plots
- $nfwp$: Number of factorial whole plots
- $nrpw$: Number of rows per whole plot
- nwx: Number of quantitative whole plot factors
- method: Which of the four approaches will be used

These values are not selected by the practitioner, they are found by observing the selected initial design that was chosen for the experiment.

The numbering for the method follows the same scheme as in section 4: 1=doubling the design, 2=pseudo-partitioning, 3=splitting the whole-plot, 4=adding a split-plot factor. All of these inputs are relative to the $RS\,(i,0;k,0)$ base design and not the design you are creating. The design matrix plus the five inputs are plugged into a function that will find an optimal design depending on the method that is chosen.

To demonstrate how the algorithm works, we will use two example designs. The first is the OLG-GLS equivalent design located in the Appendix in Table A.1 taken from table 3A in Parker, Kowalski, and Vining (2007). This is an $RS\,(2, 0; 2, 0)$ design with initial inputs:

- nwp: 10
- $nfwp$: 4
- $nrpw$: 4
- nwx: 2

The second is also an OLS-GLS equivalent design located in the Appendix in Table A.2 taken from Wang, Kowalski and Vining (2009) table 6. This is an $RS\,(3, 0; 3, 0)$ design with initial inputs:

- nwp: 18
- $nfwp$: 8
- $nrpw$: 4
- nwx: 3

These designs have been selected because they demonstrate both how the algorithm works and how some designs will have feasible allocations for some allocation strategies but not others.

3.4.1 Method 1: Splitting the Whole Plot

To demonstrate this method, we will start with the Parker, Kowalski, and Vining (2007) design in Table A.1. We will first describe the general approach of the algorithm and then explain in detail how each step is accomplished.

There are four main components to the algorithm: assigning the levels of the factorial points, assigning levels of the axial points, assessing D-optimality, and giving the results.

As our first concern is ensuring that there are enough points to clearly estimate a full first-order design for each level of the categorical factor, we will first assign levels to the whole-plot categorical factors one whole plot at a time. For each whole plot, we will loop through all possible assignments of the categorical factor within each whole plot where half are -1 and half are $+1$. This ensures that half of each whole plot is given to each level of Z_1.

Computationally, this is much more efficient as only a small number of assignments will be feasible. In this example, we will see that only 6 out of 1296 possible assignments are feasible. This leads to many less assignments that need to be tested for the whole design as we will not need to pair up whole plot assignments for the axial points with the discarded assignments.

For each possible assignment, the algorithm will verify if it is a feasible assignment by creating two separate design matrices: one where $Z_1 = -1$ and one where $Z_1 = +1$. Both of the matrices will be checked to make sure they are orthogonal. It will also verify that the whole-plots remain balanced. If they both are, the assignment will be deemed feasible and recorded.

In the next step, the algorithm will take all feasible assignments of Z_1 in the factorial points and, for each one, cycle through all possible assignments of Z_1 to the axial points. In the event that all runs in a whole plot are the same, the algorithm will automatically assign it a default setting as opposed to cycling through. This is because if all the runs are the same, it does not matter which are assigned $+1$ and which are assigned -1.

For each such assignment, if it is not discarded after the test, it will immediately be tested for optimality. The algorithm will determine the value of the objective function for the D-criterion and compare it to the previously deemed optimal design. If this value exceeds the current optimal, the design will be updated as the new optimal design. If the value of the objective function matches the current optimal, it will increase the count of ties for the optimal design. If the value is less than the current optimal, nothing will happen.

After all possible designs have been searched, the algorithm will give the vector Z_1 for a D-optimal assignment (the first design encountered that had D-optimality). It will also give the number of feasible designs and the number of designs that were tied for being optimal.

Now that we have the general structure of the algorithm, we will examine each part in detail for the specified design.

1. Assigning the levels of the factorial points:

The algorithm begins by creating a matrix $\mathbf{V}$ to find all unique partitions of the whole plot. This will contain all possible vectors of size $nrpw$ with half the entries equal to -1 and half the entries equal to 1 appended column-wise in a matrix. As there are $\binom{nrpw}{nrpw/2}$ ways to pick half the vector entries to be -1 (and the rest by default are $+1$), this will result in $\mathbf{V}$ being a matrix of size $nrpw \times \binom{nrpw}{nrpw/2}$.

In this design, $nrpw = 4$ so there are $\binom{4}{4/2} = 6$ arrangements of $(-1, -1, 1, 1)$ and $\mathbf{V}$ is the following 4×6 matrix:

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & -1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & -1 & 1 \\ -1 & -1 & -1 & 1 & 1 & 1 \end{bmatrix}.$$

We can see that all of these column vectors are unique and randomly assign all columns that are not multiplicative inverses are orthogonal to one another. This matrix $\mathbf{V}$ is now used to assign the categorical factor to the factorial whole plots.

For each of the $nfwp$ whole plots, the algorithm will cycle through and assign each whole plot one of the $\binom{nrpw}{nrpw/2}$ columns in $\mathbf{V}$. As there are $nfwp$ assignments to make and $\binom{nrpw}{nrpw/2}$ possibilities for each such assignment, the algorithm will test the $\binom{nrpw}{nrpw/2}^{nfwp}$ possible assignment of the categorical variable. In this example there are $6^4 = 1296$ possible such assignments.

For each of these allocations, the algorithm will then test which of them leads to a feasible assignment. Thus it must test if all of the columns are orthogonal in the factorial points and then it must test that all of the columns in the subsetted design are orthogonal. As soon as the assignment fails under any of the tests, the algorithm will stop and begin testing the next possible assignment until all 1296 assignments are exhausted.

To see how this works, let us see what the algorithm would do with the potential assignment $\mathbf{Z_1'} = [1, -1, -1, 1, 1, -1, -1, 1, -1, 1, 1, -1, -1, 1, 1, -1]$.

First it creates the design matrix $\mathbf{Z} = [\mathbf{Z}_1, \mathbf{X}]$ as shown in Table 3.10.

It first tests that all of the columns are orthogonal. If all of these columns turn out to be orthogonal, it will then sort the matrix in to the two subsetted designs $\mathbf{X}^+$ and $\mathbf{X}^-$.

The algorithm does so by going through each row of $\mathbf{Z}$ and testing whether the first entry is $Z_1 = +1$ or $Z_1 = -1$. If it is a $+1$, it will be added as a new row to $\mathbf{X}^+$ and, if it is a -1, it will be added as a new row to $\mathbf{X}^-$. As these subsetted matrices will have the entire first column as $+1$ or -1 for the Z_1 entry, the first column is deleted from both. For this design, the subsetted designs can be found in Table 3.11.

For each of these subsetted designs, the algorithm tests each pair of columns to see if they are orthogonal and if split-plot factor continues to sum to 0 within each whole-plot (i.e. that our whole-plot are balanced). If a split-factor remains constant through out an entire whole pot, we will have inadvertently created an additional whole-plot factor which is undesirable. As soon as one pair of columns fails or we lose balance in the split-plot factors, the algorithms stops testing and moves on to the next potential assignment.

If all of the columns in the subsetted designs are orthogonal, the split-plot factors in the subsetted designs will be tested to make sure they are still balanced within the whole plots. If they are not, the design will be discarded. This ensures that each whole plot gets some $+1$ and -1 but also ensures that we do not inadvertently create an additional whole-plot factor, which we will see shortly.

Z_1	X_1	X_2	x_1	x_2
1	-1	-1	-1	-1
-1	-1	-1	1	-1
-1	-1	-1	-1	1
1	-1	-1	1	1
1	1	-1	-1	-1
-1	1	-1	1	1
-1	1	-1	-1	-1
1	1	-1	1	1
-1	-1	1	-1	-1
1	-1	1	1	1
1	-1	1	-1	-1
-1	-1	1	1	1
-1	1	1	-1	-1
1	1	1	1	1
1	1	1	-1	-1
-1	1	1	1	1

TABLE 3.10. A potential assignment of Z_1 to the factorial points of PKV (2007) for splitting the whole plot.

$$\mathbf{X}^{+}= \begin{array}{cccc} X_1 & X_2 & x_1 & x_2 \\ \hline -1 & -1 & -1 & -1 \\ & & 1 & 1 \\ \hline 1 & -1 & -1 & -1 \\ 1 & -1 & 1 & 1 \\ \hline -1 & 1 & 1 & 1 \\ -1 & 1 & -1 & -1 \\ \hline 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \end{array} \quad \mathbf{X}^{-}= \begin{array}{cccc} X_1 & X_2 & x_1 & x_2 \\ \hline -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 \\ -1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & 1 & 1 & 1 \end{array}$$

TABLE 3.11. Subsetted designs for a candidate vector Z_1 for the factorial points in the PKV 2007 design in Table A.1

If a potential Z_1 passes this final test, it will be considered a candidate Z_1 and it will be added to the matrix **GoodZ** which tracks all feasible assignments for the factorial points. This matrix will then be printed.

For this design, the Z_1 that we demonstrated as well as 5 other assignments qualified as feasible out of 1296 potential designs. These are all shown row-wise in the following matrix:

$$\begin{bmatrix} 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \end{bmatrix}$$

Note that these 6 rows are all the rows that you would get by generating Z_1

through the multiplication of at least three other columns in addition to the negative version of that multiplication.

- **GoodZ[1,]**=$X_1x_1x_2$
- **GoodZ[2,]**=$X_1X_2x_1x_2$
- **GoodZ[3,]**=$X_2x_1x_2$
- **GoodZ[4,]**=-$X_2x_1x_2$
- **GoodZ[5,]**=-$X_1X_2x_1x_2$
- **GoodZ[6,]**=-$X_1x_1x_2$

We can see here that the columns that are useable to generate Z_1 conform to two main principles:

Result 3.4.1. *To generate a categorical Z_1 whole-plot factor when using the method of splitting whole plots, we must generate it from at least three other factors, one of which must be split-plot factor.*

The condition that we must take one of the factors we are using for generation to be a split-plot factor comes from a general principle of generating factors in fractional factorial split-plot designs. In order to generate a split-plot factor it must be generated from at least one other split-plot factor (Bingham and Sitter, 1999). As we want this added categorical factor to vary twice as often as the other whole-plot factors in order to split them, we must generate it as if it is a split-plot factor.

The principle that we require at least three other factors can be seen as follows. Suppose that we were to generate Z_1 from only two other columns, say X_1 and x_1. Then $Z_1 = X_1 x_1$. We notice that Z_1 will be positive exactly when X_1 and x_1 are equal and negative when they are different. This means that when $Z_1 = 1$, in the subsetted design we will have $X_1 = x_1$ and will be unable to clearly estimate those first-order effects. When $Z_1 = -1$, we will have $X_1 = -x_1$ and will again be unable to clearly estimate those first order effects. This directly violates one of the main principles for adding a categorical factor to an RSM design. We can see this in an example in Table 3.12.

It should also be noted that this is not an exhaustive listing of all possible generators that abide by these rules. The generators $X_1X_2x_1$, $X_1X_2x_2$, $-X_1X_2x_1$, and $-X_1X_2x_2$ are not feasible assignments. This demonstrates a second principle that applies when using the splitting the whole plot method. Note, we will see later that this is not applicable to the add split-plot method even though the algorithm functions mainly the same.

Result 3.4.2. *To generate a whole-plot factor by splitting the whole plot, it must be generated by at least two other split-plot factors.*

Previously we established that in order for our new categorical whole-plot factor to split the original whole plots in the quantitative factor in two, for each whole plot there must be two rows of $Z_1 = 1$ and two rows of $Z_1 = -1$. This will not happen if we only generate it with other whole-plot factors that vary at the same rate. Thus, we must have at least one split-plot factor in the generator.

Suppose we tried to generate Z_1 using n whole plot factors but exactly one split-plot factor. Then we have $Z_1 = X_1X_2...X_nx_1$.

X=

$Z = X_1 x_1$	X_1	X_2	x_1	x_2
1	-1	-1	-1	-1
-1	-1	-1	1	1
-1	1	-1	-1	-1
1	1	-1	1	1
-1	-1	1	1	1
1	-1	1	-1	-1
1	1	1	1	1
-1	1	1	-1	-1

$\mathbf{X}^{-}$=

X_1	X_2	x_1	x_2
-1	-1	1	1
1	-1	-1	-1
-1	1	1	1
1	1	-1	-1

$\mathbf{X}^{+}$=

X_1	X_2	x_1	x_2
-1	-1	-1	-1
1	-1	1	1
-1	1	-1	-1
1	1	1	1

TABLE 3.12. Generating the WP factor Z_1 through aliasing X_1 and x_1 in the factorial points in the PKV 2007 design in Table A.1. In $\mathbf{X}^{+}$ we have $X_1 = x_1$ and in $\mathbf{X}^{-}$ we have $X_1 = -x_1$.

When we take the partitioned designs, for $\mathbf{X}^+$ we would have $I = X_1X_2...X_nx_1$ where I is the identity column. This would imply that $x_1 = X_1X_2...X_n$. As such, it would not vary at the whole-plot level and we will have inadvertently created a whole-plot factor. As the whole plots are no longer balanced at the split-plot level, it will be deemed infeasible when it is being verified by the algorithm.

2. Assigning the levels of the axial points:

Now that the choices for assignment of the categorical factor for the factorial points has been narrowed down greatly, the algorithm will attempt to assign the categorical factor to the axial points. For all of those assignments that are considered feasible, it will then test for optimality.

Commonly in RSM split-plot designs, in the axial points, the runs are identical within many of the whole plots. In the case where runs are identical, it is of no consequence which ones get +1 and which ones get -1 as the assignments will all result in the same "re-arranged" design once we sort them back in to whole plots after splitting them.

The algorithm will first check each axial run whole plot to decide which ones have all runs the same. Once identified, the algorithm will note that those runs should always be assigned $\mathbf{v_1}$ for the added Z_1.

For each row of **GoodZ**, the algorithm will select one of the assignments from **V** for each of the $nwp - nfwp$ remaining axial point whole plots that were not flagged previously as being constant for the entire whole plot. As there are $\binom{nrpw}{nrpw/2} = 6$ (equal to the number of columns in **V**) choices for each whole plot where the runs are unequal, and only one whole plot varies between runs, we have 6 choices to cycle through on this design for assigning Z_1 for the axial points.

For each of these assignments, the algorithm will first check to make sure the design matrix including Z_1 is still orthogonal. If it is not, the assignment will be discarded. If the algorithm deems it to be a feasible assignment, it will be tested for optimality.

3. Assessing for optimality:

For the design matrix $\mathbf{Z}$ under consideration (again, this is the original $\mathbf{X}$ design matrix plus the candidate assignment of Z_1), the algorithm calculates the objective function for the D-criterion which we will describe in more detail in a later section.

If this is the first design considered, it will note that value of the objective as currently optimal, save the Z_1 column and increase the count on both the number of eligible designs and the number of optimal designs.

If this is not the first design considered, it will check its objective function value against the one that is currently registered as optimal.

For designs with a value of the objective function that is less than the current optimal design, the design will be discarded but the count on the number of good designs will increase by one.

For designs with a value that is equal to the currently optimal design, the design will be discarded but both the count for the number of good designs and the number of optimal designs will increase by one.

For designs with a value that exceeds the currently optimal design, the previous assignment will be discarded and the new proposed Z_1 will be saved as the currently optimal assignment. The count for the number of good designs will increase by one and the count for the number of optimal designs will reset to one.

4. Getting the results:

At the end, the algorithm will give the first Z_1 that leads to an optimal design as well as the number of optimal and the number of eligible designs.

$$\begin{aligned} Z_1' &= [1, -1, -1, 1, 1, -1, -1, 1, -1, 1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, \\ &\quad 1, -1, 1, -1, 1, -1, 1, -1, 1, 1, -1, -1, 1, -1, 1, -1] \end{aligned}$$

If we only consider one possible assignment for the axial whole plots where the runs are all same, there are $6^5 = 7776$ possible assignments of Z_1 in this design. This is because there are 5 whole plots (4 factorial and one added) where the runs vary and there are 6 choices of assignment per whole plot that varies. Of those, $12/7776 = 0.0015$ were eligible and the same 12 were all optimal.

However, the algorithm identified just six whole plot assignments for the factorial points in the first part of the algorithm so there were really only $6 \times 6 = 36$ designs under consideration (the 6 possible factorial point assignments times 6 choices for the one axial whole plot that varied). Considering only those designs, then $12/36 \approx 0.33$ are eligible and the same 12 are still optimal.

Now, to see a design where we cannot use this method, let us look at the design in Table A.2. Here there are whole plots with 4 sub-plots per whole plot. What is of particular note for this design, is that there are three split-plot factors.

As the whole plots have four sub-plots per whole plot, we note that the vector $\mathbf{V}$ will remain the same as:

$$V = \begin{bmatrix} 1 & 1 & -1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & -1 & 1 \\ -1 & -1 & -1 & 1 & 1 & 1 \end{bmatrix}.$$

If we examine the columns for our possible assignments, we note that at most three of them will ever be pairwise orthogonal as, letting $\mathbf{v_i} = \mathbf{V}[, i]$, we have $\mathbf{v_1} = \mathbf{v_6}$, $\mathbf{v_2} = \mathbf{v_5}$, and $\mathbf{v_3} = \mathbf{v_4}$.

As every sub-plot is already made up of three of such $\mathbf{v}_i$'s, any $\mathbf{v}_i$ that you attempt to assign to the whole-plot categorical factor will be completely aliased with one of the split-plot factors within the whole plot. That is, it will either be the same as one of the split-plot factors or it will be equal to the negative of the split-plot factor.

Thus, when this design is entered through the algorithm for the splitting the whole plot method, it will give the statement "No eligible designs".

Result 3.4.3. *If a feasible assignment of Z_1 exists for assigning a whole-plot categorical factor by splitting the whole plots, then we must have $\binom{nrpw}{nrpw/2}/2 > (nsx)$, where nsx is the number of split-plot quantitative factors.*

Letting $n(\mathbf{V})$ denote the number of columns in $\mathbf{V}$, there will always be $n(\mathbf{V})/2$ orthogonal vectors for whole plot assignment, where $\binom{nrpw}{nrpw/2} = n(\mathbf{V})$. We must assign the levels of the categorical whole-plot factor in such a way that it is orthogonal from the levels of any of the split-plot factors in the subplot.

If they were not orthogonal and Z_1 was a multiple of one of the columns, there would be two issues. Firstly, the design would not be orthogonal. But, more

X_1	X_2	X_3	x_1	x_2	x_3
-1	-1	-1	-1	-1	-1
			1	1	-1
			1	-1	1
	-		-1	1	1

$\Longrightarrow$

Z_1	X_1	X_2	X_3	x_1	x_2	x_3
1	-1	-1	-1	-1	-1	-1
1				1	1	-1
-1	-1	-1	-1	1	-1	1
-1				-1	1	1

TABLE 3.13. An example where adding a categorical WP factor that is linearly dependent on a subplot-factor creates an extra WP factor.

importantly, we would “accidentally” create an additional WP factor. When we re-arrange the design back into proper whole plots after assigning the categorical factor, we would find that the split-plot factor it is aliased with would only change at the whole-plot level which is not allowed. As the split-plot factors generally have smaller variance and more degrees of freedom associated with their respective error term, it is undesirable to create an extra whole-plot factor when it is not necessary.

We can see an example of this in Table 3.13. In the whole plot on the left we can see there are 3 whole-plot and 3 sub-plot factors. When we go to add Z_1, as there are only $\binom{6}{2}/2 = 3$ orthogonal assignments, we are forced to add an assignment that is completely aliased with one of the split-plot factors. In this particular assignment, we see $Z_1 = -x_3$. When we redistribute the whole plots (creating two whole plots instead of one), we notice that x_3 now only varies at the whole-plot level and is no longer a split-plot factor.

3.4.2 Method 2: Pseudo-Partitioning

To demonstrate this method we will start with the WKV (2009) design in Table A.2. The general approach is to first add the Z_1 to the factorial points through partitioning and then to the axial points by doubling those whole plots after which the algorithm will assess optimality.

The algorithm begins by assigning the value of Z_1 to the factorial whole plots. As every run in each whole plot gets the same value of Z_1, half of the factorial whole plots will be assigned $Z_1 = 1$ and the other half will be assigned $Z_1 = -1$.

Each possible assignment will be tested to make sure that the assignment is eligible. In the subsetted designs we verify that the columns are orthogonal to ensure we can estimate the first-order design clearly. Each eligible design will be saved.

Then, as we are doubling the axial points, for each eligible assignment in the factorial points, there is only one possible assignment of the axial points. One by one, we will append the eligible assignments of Z_1 to the factorial whole plots and then add a second copy of all axial whole plots to these designs. One copy of each each axial whole plot will get $Z_1 = 1$ and the other half will get $Z_1 = -1$.

For each of these completed designs, we know they will be eligible as we are merely adding additional runs so we do not need to verify for eligibility. As a result, the number of overall eligible designs will be equal to the number of eligible factorial whole-plot assignments.

As before, each design will have its D-criterion computed. The assignment with the highest D-criterion will be saved and the number of ties will be computed.

Now that we have the general structure of the algorithm, we will examine each part in detail for the specified design.

1. Assigning the levels of the factorial points:

The algorithm begins by making a matrix, $WPsettings$, of all possible assignments of Z_1 to the whole plots. If there are $nfwp$ factorial whole plots then half of the whole plots ($nfwp/2$) will receive $Z_1 = +1$ and half will receive $Z_1 = -1$. As all the runs within each whole plot receive the same value, we can pick the value of Z_1 for the entire whole plot at once.

The algorithm will create a vector with $(nfwp/2)$, 1's and $(nfwp/2)$, -1's. It will then create the matrix $WPsettings$ with all possible permutations of that vector in the columns. There will be $\binom{nfwp}{nfwp/2}$ columns/permutations in this matrix as we are choosing half to be -1 and be default the others must be 1.

In our example there are 8 factorial whole plots so there will be $\binom{8}{4} = 70$ possible assignments of Z_1. The matrix $WPsettings$ appears as:

$$\begin{bmatrix} 1 & 1 & 1 & \cdots & -1 & -1 & -1 \\ 1 & 1 & 1 & \cdots & -1 & -1 & -1 \\ 1 & 1 & -1 & \cdots & 1 & -1 & -1 \\ 1 & -1 & 1 & \cdots & -1 & 1 & -1 \\ -1 & 1 & 1 & \cdots & -1 & -1 & 1 \\ -1 & -1 & -1 & \cdots & 1 & 1 & 1 \\ -1 & -1 & -1 & \cdots & 1 & 1 & 1 \\ -1 & -1 & -1 & \cdots & 1 & 1 & 1 \end{bmatrix}$$

One by one, the algorithm will test the assignments in $WPsettings$ for eligibility. Suppose the algorithm was testing column $\mathbf{v}_i = WPsettings[, i]$. Starting at the first row in the factorial points, the first $nrpw$ rows will all receive the value in $\mathbf{v}_i[1]$. The next $nfwp$ rows will receive the value in $\mathbf{v}_i[2]$, etc, until all of the factorial whole-plot rows have received an assignment. This creates a candidate design $\mathbf{Z} = [\mathbf{v}_i, \mathbf{X}]$.

For example, using column 1 in our sample design, as there are $nrpw = 4$ rows per whole plot and $\mathbf{v}_1[1] = \mathbf{v}_1[2] = \mathbf{v}_1[3] = \mathbf{v}_1[4] = 1$ and $\mathbf{v}_1[5] = \mathbf{v}_1[6] = \mathbf{v}_1[3] = \mathbf{v}_1[8] = -1$, the first $4 \times 4 = 16$ rows would be assigned $Z_1 = 1$ and the next $4 \times 4 = 16$ rows would be assigned $Z_1 = -1$. The candidate design this creates can be seen in Table 3.14.

For each such assignment, the algorithm will check its eligibility. It will partition the candidate design matrix into two matrices $\mathbf{X}^+$ and $\mathbf{X}^-$ that contain the rows where $Z_1 = 1$ and $Z_1 = -1$, respectively. First it will test that all of the columns sum to 0 in both matrices. If this is not true, the candidate Z_1 will be discarded. If this is true, both of these matrices will be tested to make sure that all of the columns are orthogonal. If any of the columns fail orthogonality, the candidate assignment Z_1 will be discarded. If all columns are orthogonal, the Z_1 will be saved in the matrix **GoodZ**

In our example, the subsetted designs from the assignment in Table 3.14 will be the matrices in Table 3.15.

In this example, **GoodZ** consists of two possible assignments which are the multiplicative inverse of one another.

$$\mathbf{GoodZ} = \begin{bmatrix} 1 & 1 & 1 & 1 & -1 & -1 & -1 & \dots & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 & 1 & 1 & \dots & -1 & -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$

We note that for this design, the only eligible designs are those made through generation following the general rules for generating whole-plot factors and result 3.4.1.

Z_1	X_1	X_2	X_3	x_1	x_2	x_3
1	-1	-1	-1	-1	-1	-1
				1	1	-1
				1	-1	1
				-1	1	1
1	1	-1	-1	1	-1	-1
				-1	1	-1
				-1	-1	1
				1	1	1
1	-1	1	-1	1	-1	-1
				-1	1	-1
				-1	-1	1
				1	1	1
1	1	1	-1	-1	-1	-1
				1	1	-1
				1	-1	1
				-1	1	1
-1	-1	-1	1	1	-1	-1
				-1	1	-1
				-1	-1	1
				1	1	1
-1	1	-1	1	-1	-1	-1
				1	1	-1
				1	-1	1
				-1	1	1
-1	-1	1	1	-1	-1	-1
				1	1	-1
				1	-1	1
				-1	1	1
-1	1	1	1	1	-1	-1
				-1	1	-1
				-1	-1	1
				1	1	1

TABLE 3.14. Candidate assignment of Z_1 to the factorial points in WKV 2009 design.

$\mathbf{X}^{+}=$

X_1	X_2	X_3	x_1	x_2	x_3
-1	-1	-1	-1	-1	-1
			1	1	-1
			1	-1	1
			-1	1	1
1	-1	-1	1	-1	-1
			-1	1	-1
			-1	-1	1
			1	1	1
-1	1	-1	1	-1	-1
			-1	1	-1
			-1	-1	1
			1	1	1
1	1	-1	-1	-1	-1
			1	1	-1
			1	-1	1
			-1	1	1

$\mathbf{X}^{-}=$

X_1	X_2	X_3	x_1	x_2	x_3
-1	-1	1	1	-1	-1
			-1	1	-1
			-1	-1	1
			1	1	1
1	-1	1	-1	-1	-1
			1	1	-1
			1	-1	1
			-1	1	1
-1	1	1	-1	-1	-1
			1	1	-1
			1	-1	1
			-1	1	1
1	1	1	1	-1	-1
			-1	1	-1
			-1	-1	1
			1	1	1

TABLE 3.15. Partioned matrices $\mathbf{X}^{+}$ and $\mathbf{X}^{-}$ for candidate assignment of Z_1 to the factorial points in WKV 2009 design.

We know that whole-plot factors can only be generated from other whole-plot factors and the result tells us that we require at least three other factors to generate from. As there are exactly three whole-plot factors, we can only use $X_1X_2X_3$ and $-X_1X_2X_3$ to do the generation.

Up until this point, we have seen only designs where all feasible designs are done by generation. However, we will see further examples where there are other feasible assignments beyond generation.

2. Assigning the levels of the axial points:

For each row in **GoodZ**, the algorithm will now assign the remaining values of Z_1. In pseudo-partitioning this is done by doubling the remaining whole plots.

For all of the rows in the original design matrix that are not from the factorial points, the algorithm will append a second copy of all the rows (in the same order to maintain the whole-plot structure) to the end of the design matrix. The factorial point rows will be given values of Z_1 according to the current row of interest in **GoodZ**, the first copy of the axial point rows will be assigned $Z_1 = 1$ and the second copy of the axial points will be assigned $Z_1 = -1$.

Note that this obviously implies there is only one possible assignment of the axial points and the total number of designs under consideration in this situation will be equal to the number of columns in **GoodZ**.

3/4. Assessing for optimality and getting the results:

For each design $\mathbf{Z}$ created in step 2, the algorithm will calculate the objective function for the D-criterion and, as in the splitting the whole plot method, it will sort through them to keep the first design with the highest optimality criterion and count how many others are optimal and how many others are eligible. The algorithm will output those statistics as well as the Z_1 column for the optimal design.

In our example, as there is essentially only one candidate design (one candidate plus its multiplicative inverse), both of the designs have the same value of the objective function and are both optimal.

While pseudo-partitioning is an option for this design, we can see an example where this is not the case with the PKV 2007 design from the splitting the whole plot example in Table A.1.

In this design there are two whole-plot factors and four factorial whole plots. Looking just at the levels of the whole-plot factors in the factorial points we have

X_1	X_2
-1	-1
1	-1
-1	1
1	1

When we go to assign the levels of Z_1 to these whole plots, as we are assigning a vector of length four, we have already established that we have three possible choices (excluding multiplicative inverses which either be orthogonal or not, the same as its "pair"): (-1, -1, 1, 1), (-1, 1, 1, -1), and (-1, 1, -1, 1).

Option 1 and option 3 are the same as X_1 and X_2 respectively so they are not valid choices if we would like our columns orthogonal. This leaves us with option 2. Suppose we were to use option 2. When we go to partition the design we would get the issue in Figure 3.16.

Z_1	X_1	X_2
-1	-1	-1
1	1	-1
1	-1	1
-1	1	1

$\downarrow$

$$\mathbf{X}^{-} = \begin{array}{cc} X_1 & X_2 \\ \hline -1 & -1 \\ 1 & 1 \end{array} \quad \mathbf{X}^{+} = \begin{array}{cc} X_1 & X_2 \\ \hline 1 & -1 \\ -1 & 1 \end{array}$$

TABLE 3.16. An example of a design where we cannot apply pseudo-partitioning.

As in the partitioned designs we have $X_1 = X_2$ in $\mathbf{X}^-$ and $X_1 = -X_2$ in $\mathbf{X}^+$, we would not be able to clearly estimate the first-order design at each level of Z_1 and we do not have any valid way of assigning Z_1.

This demonstrates that for some designs, only one of the two methods other than doubling the whole plot may be possible. Thus, it may not just be practical issues that dictate which method to use but there may be mathematical constraints as well if you have a high need to use a particular design as your base design.

3.4.3 Method 3: Adding a Split-Plot Factor

For the case where we are adding a split-plot factor, the algorithm works nearly identically to the situation where we are adding a whole-plot factor by splitting the whole plot. The solution, however, will not be the same. We are now creating whole plots of the same size so that the objective function of the D-optimality criterion will be calculated differently. Also, as we are not able to accidentally create an additional whole-plot factor, there are less reasons why a design may be infeasible.

The algorithm follows the same general pattern as splitting the whole plots: first it will attempt to assign levels of z_1 to the factorial whole plot by cycling through all of possible assignments within a whole plot for each whole plot. For each assignment, the algorithm will check if it is feasible (whether the partitioned designs are balanced and orthogonal). Then, for each axial whole plot where the runs are the same, it will be assigned a default setting. For each axial whole plot where at least one run is different, all possible assignments will be cycled through. Afterwards, each eligible design will be tested to find out which is optimal and how many designs tie for being optimal.

We can see how the algorithm works in detail using the PKV 2007 example.

1. Assigning the factorial points

In this design we recall that there are four whole plots with four runs each. The algorithm begins by finding all permutations of the vector of size *nrpw* with half the entries as $z_1 = 1$ and half the entries as $z_1 = -1$. As before, this vector is:

$$V = \begin{bmatrix} 1 & 1 & -1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & -1 & 1 \\ -1 & -1 & -1 & 1 & 1 & 1 \end{bmatrix}$$

The algorithm will cycle through all possible combinations of the columns of $\mathbf{V}$ for each of the whole plots. As there are 4 whole plots with 6 options each, there are $6^4 = 1296$ combinations to consider.

For each candidate z_1, made up of 4 columns of $\mathbf{V}$ with replacement, the algorithm will partition the design into $\mathbf{X}^+$ and $\mathbf{X}^-$ as before.

For both $\mathbf{X}^+$ and $\mathbf{X}^-$ the algorithm will test whether the columns are orthogonal and whether the columns sum to 0. In contrast to the splitting the whole plot function, the algorithm will not check for balance within the whole plots in the subsetted designs. It is impossible to inadvertently create a whole-plot factor as the whole plots are remaining the exact same size and, by construction, half of each whole plot gets $z_1 = 1$ and the other half gets $z_1 = -1$.

As such, in our example, despite assigning z_1 the same way we assign Z_1 in the splitting the whole-plot example, we will have 10 feasible assignments of z_1 as opposed to 6. We are now allowed to use the generators $z_1 = X_1X_2x_1$, $z_1 = X_1X_2x_2$, $z_1 = -X_1X_2x1$, and $z_1 = -X_1X_2x_2$.

The algorithm will then record all of the z_1's that are feasible in a matrix **GoodZ** where the rows represent the feasible assignments for z_1. For the PKV example we have:

$$\mathbf{GoodZ} = \begin{bmatrix} 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \end{bmatrix}$$

2. Assigning the axial whole plots:

After finding all of the feasible assignments for the factorial whole plots, the algorithm will then find all possible assignments of the axial whole plots.

As before, it will begin by identifying which of the whole plots have all runs the same. Those whole plots will be flagged and by default, those whole plots will receive the first column in $\mathbf{V}$ as the settings for z_1.

For the remaining whole plots, the algorithm will cycle through all combinations of columns in $\mathbf{V}$ to assign to those whole plots.

In our example there are 6 axial whole plots, one of which has different runs. All 6 of those combinations will be matched up with all 10 of the feasible assignments in **GoodZ**.

Each of those assignments will be tested to make sure they are feasible by checking that the columns remain orthogonal. Of the 60 possible combinations in PKV 2007 being tested, 20 of them are flagged by the algorithm as being feasible.

3/4. Testing for optimality and getting the results

For each design that is flagged as feasible, as before, the algorithm tests for the value of the objective function for the D-criterion. It will save a design if it beats the previously recorded best design, discard if it is less than the previously recorded design, and increase the tie counter if it ties for optimality.

As with the other settings the algorithm will print the value of the objective function of the max design, give the recorded optimal z_1 and state how many designs were feasible and how many were optimal.

In the PKV design there were 20 considered feasible and all of them were considered optimal. This is $20/(6^4 \times 6) = 0.0026\%$ of all designs that could have been considered (there were 6^4 possible factorial whole plot assignments matched up with 6 possible axial point assignments).

3.4.4 Assignment Beyond Generation

Up until now, we have seen designs where we are doing no better than generating factors with certain additional constraints on top of the usual constraints for generating factors in split-plot designs. Looking at the WKV 2009 design, we can see an option where there are more feasible designs available than by simply generating.

Recall in this design that we have 8 factorial whole-plots with four runs per whole plot. Thus, to assign a split-plot categorical factor, we are assigning levels of z_1 within each whole plot from the six column-vectors in $\mathbf{V}$,

$$V = \begin{bmatrix} 1 & 1 & -1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & -1 & 1 \\ -1 & -1 & -1 & 1 & 1 & 1 \end{bmatrix}.$$

In total there will be 8^6 possible assignments of the split-plot factor z_1 (half of which will be the multiplicative inverse of the others). Running the algorithm reveals that there are 90 possible feasible assignments of z_1. How many would be available through generation?

When we go to generate we would choose to generate with between 0 to 3 whole-plot factors and 0 to 3 split-plot factors. Our constraints are that we require at least one split-plot factor to generate with and we must generate with at least three other factors. We can count out our options in Table 3.17.

There are 41 unique possible combinations for generators plus we need to consider the negative of all the generators for a total of $41 \times 2 = 82$ choices. With 90 feasible z_1's this means that we have eight feasible options that are not from generation. As we increase our number of columns in $\mathbf{V}$ or we add more whole plots, we increase the number of options and checking through all of our options becomes essential beyond just using generating techniques.

From this we can draw the conclusion that designs with fewer factors or larger numbers of rows per whole plot may need us to consider designs beyond generation. But, for small designs, it may be appropriate to simply select the design based on generating with the largest number of allowable factors according to our rules.

3.4.5 What is an Optimal Design?

In this algorithm, the optimality criterion used is D-optimality. Historically, D-optimality was one of the first criteria used for response surface methodology for designs with categorical factors in Draper ad John (1988) and continues to be utilized since in papers such as Cook and Nachtsheim (1989), Trinica and Gilmour (2001), and Wu and Ding (1998). It is also used in split-plot RSM papers such as Kowalski, Cornell, and Vining (2002), Macharia and Goos (2010), Jones and Goos (2012a), Jones and Goos (2012b), Lin and Yang (2015), and Nguyen and Pham (2015). While it is not the only criterion that could have been selected, it is a good first treatment as it is comparable to these many other papers on split-plot RSM designs.

# WP	# SP	Feasible?	# options
0	0	$\times$	-
0	1	$\times$	-
0	2	$\times$	-
0	3	$\checkmark$	1
1	0	$\times$	-
1	1	$\times$	-
1	2	$\checkmark$	9
1	3	$\checkmark$	3
2	0	$\times$	-
2	1	$\checkmark$	9
2	2	$\checkmark$	9
2	3	$\checkmark$	3
3	0	$\times$	-
3	1	$\checkmark$	3
3	2	$\checkmark$	3
3	3	$\checkmark$	1
			$\sum = 41$

TABLE 3.17. A count of the number of generators for z_1 in PKV 2007 up to multiplicative inverses

In experimental design, for a given design matrix $\mathbf{X}$, the moment matrix is given by

$$|\mathbf{M}| = \frac{\mathbf{X}'\mathbf{X}}{N} \tag{3.1}$$

where N is the number of runs in the design. The design matrix $\mathbf{X}$ is a listing of the columns for not only our main effects but columns for all interaction terms included in the model as well as an intercept column.

By Myers and Montgomery (1995), the determinant is

$$|\mathbf{M}| = \frac{\mathbf{X}'\mathbf{X}}{N^p} \tag{3.2}$$

where p is the number of parameters. They state that "the determinant of $\mathbf{X}'\mathbf{X}$ is inversely proportional to the square of the volume of the confidence region on the regression coefficients". Thus we can see the connection Myers and Montgomery make. To achieve precision in our estimation of the coefficients, we would like a small confidence region. If the confidence region and $|\mathbf{M}|$ are inversely proportional then we would like to maximize $|\mathbf{M}|$. Thus "a D-optimal design is one in which $|\mathbf{M}| = |\mathbf{X}'\mathbf{X}|/n^p$ is maximized" (Myers and Montgomery, 1995).

Thus, after finding the designs that are feasible, our algorithm will calculate all of the D-criterion values and select the design that has maximized $|\mathbf{M}|$.

In a split plot design, the information matrix is given by

$$\mathbf{M} = \mathbf{X}'\mathbf{V}^{-1}\mathbf{X} \tag{3.3}$$

where $\mathbf{V}$ is the variance-covariance matrix given by

$$\mathbf{V} = \sigma_\epsilon^2\mathbf{I}_n + \sigma_\gamma\mathbf{Z}\mathbf{Z}'. \tag{3.4}$$

Here $\mathbf{Z}$ is an $n \times b$ matrix where b is the number of whole plots in the design. The entry $\mathbf{Z}[i, j] = 1$ if row i belongs to whole plot j.

To calculate the D-criterion we assume that we have equal variances in the whole-plot and split-plot error. This assumption is necessary as we cannot practically know the true variances and $\mathbf{V}$ is much harder to calculate without this assumption. Thus we have,

$$|\mathbf{M}| = |\mathbf{X}'(\sigma_\epsilon^2\mathbf{I}_n + \sigma_\gamma\mathbf{Z}\mathbf{Z}')^{-1}\mathbf{X}| = \frac{1}{(\sigma^2)^n}|\mathbf{X}'(\mathbf{I}_n + \mathbf{Z}\mathbf{Z}')^{-1}\mathbf{X}|.$$

As $(\sigma^2)^n$ is a positive constant, maximizing $|\mathbf{M}|$ is equivalent to maximizing $|\mathbf{X}'(\mathbf{I}_n + \mathbf{Z}\mathbf{Z}')^{-1}\mathbf{X}|$. Instead of calculating $|\mathbf{M}|$ directly to maximize, the algorithm will calculate $|\mathbf{X}'(\mathbf{I}_n + \mathbf{Z}\mathbf{Z}')^{-1}\mathbf{X}|$ instead. This proxy for $|\mathbf{M}|$ is called the objective function.

In the optimality section of the algorithm, as all potential designs for any particular run of the algorithm will have the same structure, the algorithm will first create the matrix $\mathbf{Z}$. The number of rows in the candidate design will be determined as well as the number of whole plots. If the doubling the whole plot method is being used, the number of whole plots will be double the original number. If pseudo-partitioning is being used, the number of runs in the design will also be increased as we are doubling the number of axial points.

Using this information, the algorithm will create the matrix $\mathbf{Z}$ by first creating a blank $n \times b$ matrix. Then, starting from the $\mathbf{Z}[1,1]$ entry, the first $nrpw$ rows in column in 1 will get a 1. Then it will move to column 2 and fill in the next $nrpw$ rows in column 2 with a 1. This will continue until the entire matrix has been filled.

After creating $\mathbf{Z}$, the algorithm will calculate $|\mathbf{X}'(\mathbf{I}_n+\mathbf{Z}\mathbf{Z}')^{-1}\mathbf{X}|$ for each candidate assignment. As we went through in the detailed explanation of the algorithm, each candidate matrix will have this number calculated. If it is less than the current max, the design gets discarded. If it is equal to the current max, the design gets discarded but we increase the count of optimal designs. If it exceeds the current max, we will discard all previously saved information on the optimum and save the new assignment while reseting the count on the number of optimal designs to 1.

For example, consider the PKV (2007) design from Table A.1 that we used for the splitting the whole-plot example. After running this design through the algorithm, there were 12 feasible rows that could be appended for splitting the whole plot, all of which have the same value of the objective function for our D-criterion. Let us see what would happen for the first considered assignment,

$$\begin{aligned} Z_1' &= 1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,-1,1,1,-1,1,1,-1,-1,1,1,-1,-1 \\ &\quad 1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1. \end{aligned}$$

This column needs to be appended on as the left column to the design as it appears Table A.1. The result of this can be seen in Table 3.18. Then the design should be resorted into whole plots as splitting the whole plot may have put the rows in the wrong order. This new matrix can be found in Table 3.19.

Next the algorithm will find the design matrix for fitting the model in equation 2.3. It starts by finding the columns $I(1)$ and $I(-1)$ which denote when $z_1 = +1$ and when $z_1 = -1$ respectively. Then it will append $X_1(1)$ (equal to $X_1 \cdot I(1)$), $X_1(-1)$,. . . , $x_2(1)$, and $x_2(-1)$. After that will appear all of the columns for the second-order factor effects and the interaction effects. The design matrix for this design can be found in Table 3.20.

Next, to get the information matrix, the algorithm will calculate $\mathbf{ZZ'}$. Here we have a $n \times n = 40 \times 40$ block diagonal matrix with 20 pairs of 1's down the diagonal as we have 20 whole plots of size 2. The matrix will look like:

$$\begin{bmatrix} 1 & 0 & \dots & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix}$$

Now the algorithm will compute the determinant of $|\mathbf{X}'(\mathbf{I}_n + \mathbf{ZZ}')^{-1} cvvc\mathbf{X}|$ which is our objective function for the D-criterion.

Note that, because the D-criterion requires the design matrix, it is model specific. All of the tables were created assuming we were fitting a complete second-order

z_1	X_1	X_2	x_1	x_2	z_1	X_1	X_2	x_1	x_2
1	-1	-1	-1	-1	1	$\sqrt{2}$	0	0	0
-1	-1	-1	1	-1	1	$\sqrt{2}$	0	0	0
-1	-1	-1	-1	1	-1	$\sqrt{2}$	0	0	0
1	-1	-1	1	1	-1	$\sqrt{2}$	0	0	0
1	1	-1	-1	-1	1	0	-$\sqrt{2}$	0	0
-1	1	-1	1	1	1	0	-$\sqrt{2}$	0	0
-1	1	-1	-1	-1	-1	0	-$\sqrt{2}$	0	0
1	1	-1	1	1	-1	0	-$\sqrt{2}$	0	0
-1	-1	1	-1	-1	1	0	$\sqrt{2}$	0	0
1	-1	1	1	1	1	0	$\sqrt{2}$	0	0
1	-1	1	-1	-1	-1	0	$\sqrt{2}$	0	0
-1	-1	1	1	1	-1	0	$\sqrt{2}$	0	0
-1	1	1	-1	-1	1	0	0	$-\sqrt{2}$	0
1	1	1	1	1	1	0	0	$\sqrt{2}$	0
1	1	1	-1	-1	-1	0	0	0	$-\sqrt{2}$
-1	1	1	1	1	-1	0	0	0	$\sqrt{2}$
1	-$\sqrt{2}$	0	0	0	1	0	0	0	0
1	-$\sqrt{2}$	0	0	0	1	0	0	0	0
-1	-$\sqrt{2}$	0	0	0	-1	0	0	0	0
-1	-$\sqrt{2}$	0	0	0	-1	0	0	0	0

TABLE 3.18. Appending the D-optimal assignment of z_1 for splitting the whole-plot to PKV (2007) from Table A.1.

z_1	X_1	X_2	x_1	x_2	z_1	X_1	X_2	x_1	x_2
1	-1	-1	-1	-1	1	2	0	0	0
		-1	1	1			0	0	0
1	1	-1	-1	-1	-1	2	0	0	0
		-1	1	1			0	0	0
1	-1	1	1	-1	1	0	-2	0	0
		1	-1	1			-2	0	0
1	1	1	1	-1	-1	0	-2	0	0
		1	-1	1			-2	0	0
-1	-1	-1	1	-1	1	0	2	0	0
		-1	-1	1			2	0	0
-1	1	-1	1	-1	-1	0	2	0	0
		-1	-1	1			2	0	0
-1	-1	1	-1	-1	1	0	0	-2	0
		1	1	1			0	2	0
-1	1	1	-1	-1	-1	0	0	0	-2
		1	1	1			0	0	2
1	-2	0	0	0	1	0	0	0	0
		0	0	0			0	0	0
-1	-2	0	0	0	-1	0	0	0	0
		0	0	0			0	0	0

TABLE 3.19. The D-optimal design for adding a whole-plot by splitting the whole-plot for the PKV (2007) design in Table A.1.

	Ip	Im	X1p	X1m	X2p	X2m	x1p	x1m	x2p	x2m	X1X1	X2X2	x1x1	x2x2	X1X2	X1x1	X1x2	X2x1	X2x2	x1x2
[1,]	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[2,]	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[3,]	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[4,]	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[5,]	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[6,]	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[7,]	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[8,]	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[9,]	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[10,]	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[11,]	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[12,]	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[13,]	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
[14,]	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
[15,]	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
[16,]	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
[17,]	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
[18,]	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
[19,]	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
[20,]	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
[21,]	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
[22,]	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
[23,]	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
[24,]	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
[25,]	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
[26,]	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
[27,]	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
[28,]	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
[29,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
[30,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
[31,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
[32,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
[33,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
[34,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
[35,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
[36,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
[37,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
[38,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
[39,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
[40,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

TABLE 3.20. The design matrix for the PKV (2007) design from Table A.1 with D-optimal assignment for adding a whole-plot by splitting the whole-plot. After a coefficient, "p" denotes (1) and "m" denotes (-1).

model. As the model is written in the linear form, there is not a well defined interpretation to the effect on the shape of the response surface for any particular second-order term. Thus, it would be unusual to know in advance that a model term, say β_{11}, would not be significant as a shaping parameter. For this reason, the program currently does not consider reduced models when finding the D-optimal design though it could be made to do so.

In future iterations of this algorithm, it would not be difficult to program in different optimality criteria that are compared by finding the maximum or minimum of a single value such as I-optimality, G-optimality or one of the other types of D-optimality considered in Wu and Ding (1998) that were discussed in the literature review. An extra input variable could be programmed that would toggle which optimality criterion to use similar to the assignment method if desired. Certainly other types of optimality criterion that do not rely on minimizing/maximizing an objective function could be incorporated to the algorithm but a substantial amount of new programming would need to be done to incorporate it in to the current code framework.

It is also worth noting that the algorithm only saves the first D-optimal design that is encountered and discards the rest. This is not to imply that all designs tied for D-optimality are completely "equal". While most split-plot RSM papers that are surveyed for this thesis only search through a single search criterion, modifications could be made to apply a second optimality criterion amongst all D-optimal designs to further select the "best" model if there was another criterion of interest.

As a point of interest, several of the designs were run through the computer software program JMP to see what the generally D-optimal design would be. The computer generated designs were unbalanced, were not guaranteed to be "good" designs at each level of the categorical factor (for example were lacking center points at each level of z) and were generally less practitioner friendly. This desire to make sure we have a proper design when collapsed over z and for each separate response surface is the motivating factor for using this method and not just relying on a computer generated design.

3.5 The Algorithm and Efficiency

In this section we will address improvements made to the code to make it more computationally efficient and current computational limits of the code with a possible solution.

In creating the algorithm, several checks and considerations were made to help with the speed of calculations for larger designs. In the factorial points, the number of designs under consideration was a mixture of both the number of whole plots and the number of runs per whole plot. Designs with large numbers of runs per whole plot can take considerably longer to run due to the size of the $\mathbf{V}$ matrix.

A large savings in computational time came from three factors: eliminating designs with poor assignment of the factorial points before attempting to assign the axial points, adding frequent breaks to the code to test for failures, and only cycling through settings in the axial points where the runs change within the axial point.

In the description of the algorithm, we saw how it would first assign the levels of Z_1 (or z_1) to the factorial points first. The number of possible assignments for the factorial whole plot, as mentioned previously, is $\binom{nrpw}{2}^{nfwp}$. While this number is quite large, there are typically relatively few of those assignments that will allow for eligible assignments. For example, in the VKMU_CCD_D24 design there are 4096 possible assignments in the whole plots of which only 140 are eligible.

As there are usually a larger number of axial whole plots, it requires much less time to test assignments in those whole plots only with the 140 eligible factorial whole-plot assignments and not all 4096 assignments.

There are other checks that have been put in place in the algorithm to help with time efficiency. There are frequent checks throughout the algorithm to decide if assignments are balanced or if they are orthogonal. Breaks have been put in to the loops to ensure that as soon as a design is flagged as ineligible, it will no longer go through any of the checks and the algorithm will proceed to the next possible assignment. The addition of the breaks in the algorithm helped to reduce the computational time by upwards of 30% depending on how many ineligible assignments there were for that particular design. Depending on the design, this savings was measured in minutes, hours, or in some cases several days for the computing time.

Typically in split-plot RSM designs, there are more added whole plots than factorial whole plots. The designs we considered, however, did not have many added whole plots where the runs varied within the whole plot. As such, the algorithm was designed to detect in which whole plots had no variation (i.e. all runs are the same) and they were assigned a default assignment of -1 for the first half of the whole plot and 1 for the second half of the whole plot in both the splitting the whole plot method and the adding a split-plot method.

If you were wanting to use a base design where there was not this homogeneity within the added whole plots, you may encounter a situation where you have a computationally infeasible number of assignments to check. In an early iteration of the algorithm with out this precaution taken, it took over a week for the algorithm to run on some of the larger sized designs (relative to the ones under consideration). Some designs failed after a week due to lack of available memory. In the current version, no designs fail because of assignment to the axial points.

The designs were run on the University of Manitoba Department of Statistics' Computational Cluster. The computational cluster consists of four Dell PowerEdge R620 servers that each have 32 CPU cores and 64 GB of memory. The computational clusters, along with the mentioned improvements to the algorithm, were able to allow me to run designs that were impossible to run on a personal computer and drastically reduced the running time of many designs.

However, there were some designs that required the storing a matrix of possible assignments of columns of $\mathbf{V}$ that had 70^8 or more rows. These designs were not able to be run on the cluster, unfortunately, due to an inability to store a matrix of this size. In the tables, these designs are denoted by a *. This means that I was unable to run an exhaustive global search. Generally, the size of the $\mathbf{V}$ matrix is the most likely indicator that a design may fail computationally.

One suggestion for these designs would be to attempt a random search of column assignments. Instead of saving a matrix with all possible assignments of $\mathbf{V}$ columns for the factorial points of Z_1 or z_1, you could set the algorithm to try N random combinations of the columns of $\mathbf{V}$ for the factorial whole plots. This may not, however, find you a feasible assignment. In 100,000,000 random assignments I was not able to find a feasible assignment for the design where $\mathbf{V}$ has 70^8 rows. For 70^8

globally possible assignments, 100000000 assignments is only $1.734665 \times 10^{-}5\%$ of all possible assignments. As the number of feasible assignments tends to be small relative to the number of possible assignments, this is not surprising that you would generally be unable to find a feasible assignment.

Another suggestion for these designs would be to follow the generation rules and see if there is an eligible assignment to be found that way. From the tables of optimal designs at the end of this chapter, we can see that most of the time all eligible assignments are optimal and, of those where that is not the case, generally half of the assignments are still optimal. This means that finding an eligible assignment is often good enough.

For example, look at design 24 (WVK_2SR_D33) found in Table B.24. We can see the 64 factorial points in Table A.3. According to the optimal design tables, there are too many combinations to do a global search for z_1 when we want to add a split-plot factor. Our rules tell us that we must generate with at least three other factors. Consider the assignment $z_1 = x_1x_2x_3$.

The first whole plot, for example, is:

```
[1,] -1.000 -1.000 -1.000 -1.000000 -1.000000 -1.000000
[2,] -1.000 -1.000 -1.000 -1.000000 -1.000000  1.000000
[3,] -1.000 -1.000 -1.000 -1.000000  1.000000 -1.000000
[4,] -1.000 -1.000 -1.000 -1.000000  1.000000  1.000000
[5,] -1.000 -1.000 -1.000  1.000000 -1.000000 -1.000000
[6,] -1.000 -1.000 -1.000  1.000000 -1.000000  1.000000
[7,] -1.000 -1.000 -1.000  1.000000  1.000000 -1.000000
[8,] -1.000 -1.000 -1.000  1.000000  1.000000  1.000000
```

The last three columns are $x_1x_2x_3$. We can see that rows 1, 4, 6, and 7 have a product of -1 and row 2, 3, 5, and 8 have a product of +1. The partitioned whole plots would be:

$\mathbf{X}^{-}=$

```
[1,] -1.000 -1.000 -1.000 -1.000000 -1.000000 -1.000000
[4,] -1.000 -1.000 -1.000 -1.000000  1.000000  1.000000
[6,] -1.000 -1.000 -1.000  1.000000 -1.000000  1.000000
[7,] -1.000 -1.000 -1.000  1.000000  1.000000 -1.000000
```

$\mathbf{X}^{+}=$

```
[2,] -1.000 -1.000 -1.000 -1.000000 -1.000000  1.000000
[3,] -1.000 -1.000 -1.000 -1.000000  1.000000 -1.000000
[5,] -1.000 -1.000 -1.000  1.000000 -1.000000 -1.000000
[8,] -1.000 -1.000 -1.000  1.000000  1.000000  1.000000
```

We can see that the columns in both matrices are orthogonal and that we have not inadvertently created any whole-plot factors. The reader can verify that this would be the case for the remaining whole plots as well. It is easy to see that using (-1, -1, -1, -1, 1 1 1 1) for the added whole plots will result in the whole matrix being orthogonal and balanced.

Thus, we have found a feasible assignment of z_1 despite being unable to do an exhaustive global search and unable to find a feasible assignment through random search. We can not say whether this assignment is optimal but it will work if needed.

It is also an option to modify the algorithm if needed to search through all possible generators to find at least the optimal assignment amongst generated designs.

As the algorithm requires a large number of computations, we can see that it was possible to find solutions that cut down the number of comparisons by instituting multiple checks with breaks and assigning the categorical factor in stages. This was not able to overcome all computational difficulties and there are still some very large designs for which the computations can not feasibly be accomplished with the given resources. We do have options, however, if it is possible to find generators that lead to feasible assignment through the generating rules we saw previously. In our tables we will focus on designs with up two 6 factors.

3.6 Tables and Designs

Now we will discuss the methods that were chosen for inclusion in the catalog, the reasoning behind excluded designed, and then explain the use of the catalogs of optimal designs available at the end of this section.

There are five categories of base designs used in the catalog:

1. VKM Unbalanced CCD method (equivalent estimation design)
2. VKM CCD method (equivalent estimation design)
3. Two-strata rotatable designs
4. CUBE designs
5. Balanced VKM Method for small composite designs (SCD)

Discussions for the format/uses of these designs and their advantages/disadvantages can be found in Chapter 2 in the literature review. The source references either the paper that catalogues the designs or the equivalent estimation catalogue of Peter Parker found at http://theses.lib.vt.edu/theses/available/etd-03302005-194026/unrestricted/Equivalent_SplitPlot_Designs.pdf.

In this initial research, we were looking for best practices in assignments and general principles for ensuring feasibility. In particular, the focus was on designs that would lend themselves to practitioners who would be choosing designs that would be easier to analyze. As designs of this nature are not currently built in to common statistical software for analysis, designs with "nice" features such as balance within whole plots and whole plots of equal size were selected to receive the first thorough treatment for design selection.

For this first iteration of the algorithm, the computer code is designed in a way that exploits that extremely balanced nature of the designs such as finding $\mathbf{V}$ by taking a vector that is literally half +1 and half -1.

Not all designs in the literature for split-plot RSM with quantitative factors limit themselves to this feature. Thus, not all possible "good" base designs can be used. Further research which extends to designs that are not balanced will require a reprogramming of the algorithm so that it is not reliant on balance.

In particular, this research currently excludes the usage of the minimum whole plot designs which have whole plots of size $2^r + 1$ for any integer r. Recall that in these designs, to avoid having to devote an entire whole plot to the center points, a center point is added to each of the whole plots. One complication is that the whole-plot sizes are an odd number. We would need to create $\mathbf{V}$ using vectors where there was one more or one less -1 than +1, taking in to consideration that we would not always want to choose the assignment with extra -1 (or +1). Addressing this issue for minimum whole plot design also opens up possibilities for other unbalanced designs.

It would also require consideration to ensure that each level of the categorical factor received center points. As they are not all in the same whole plot, we cannot be sure that it will happen automatically as it currently does in designs under consideration.

As we are always acting under the assumption that the cost of runs is at a premium, further investigation in to minimum whole plot designs is certainly worthy of future research and we will address this further in Chapter 4 for future research.

In the Parker equivalent estimation catalog, there were other designs that were not considered. Many of these were due to them having whole-plots of very small sizes (such as size 2) which makes assigning the categorical variable obvious or infeasible (such as splitting a whole plot as you would have whole plots of size 1 and it would no longer be a split-plot design). There were also some designs where the whole-plot sizes changed drastically from whole plot to whole plot so that the algorithm could not accommodate them without a complete rewrite. This did not seem a fruitful venture to pursue as many had whole plots of size 2 which, as above, would not be logical to deal with in this context.

There is a solid foundation of base designs used in the five categories in this thesis which will form a good basis for the research in this field. There is certainly potential for future research in non-regular or unbalanced designs after a thorough treatment of the balanced case.

With a grasp on the designs used, we can learn how to make use of the table. Let us use the first table grouping in subsection 3.6.1, the unbalanced VKM method CCD designs. It is titled with the construction method for the grouping of base designs and the original source for the designs used.

The first table lists all base designs used for that particular construction method. Column 1 is the design number. These numbers are assigned by me and correspond to the order the designs appear in the appendix. The next column is the name. If the designs come from the equivalent estimation catalogue of Peter Parker then they have the name given to them in that catalogue.

For example, design 1 is VKMU_CCD_D12. The complete design can be viewed in Figure B.1. For designs not from the catalogue, I have named them in a similar style where, for example, a design that has D12 in the label has one whole plot and two split-plot quantitative factors.

The next two columns are the number of quantitative whole-plot and split-plot factors respectively. Design 1 has one whole-plot and two split-plot factors so has table entries 1 and 2. The fifth column is the number of whole plots in the design. For design 1, there are 6 whole plots. The sixth column is *nrpw* which is the notation in the algorithm section for the number of rows per whole plot (this is the same thing as the number of sub-plots per whole plot). These are 4 rows per whole plot in design 1. The last column is the number of rows in the base design. In our example design there are 24. Note that this is the product of columns 6 and 4 (number of whole plots×number of rows per whole plot).

The next table catalogues optimal assignments of Z_1 when assigning the column by splitting the whole plot. They are organized by design number as found in the above table.

For design 1, in the first row, we can see the design number. This is followed by a check to indicate that there is a feasible assignment available. Under "# Eligible" it has a 4 meaning there are 4 eligible assignments of Z_1. Under "# Optimal" it has a 4 meaning that, of the four eligible assignments, four are optimal.

The second row of design 1 gives an optimal assignment of Z_1. To use it, you select the table for design 1 in Table B.1 and then append it to the left of the table from top to bottom reading the row from left to right. The final design you would get can be seen in Table 3.21.

Note, that to use this design, you would need to re-arrange the rows in to whole plots. For example, row 1 and row 4 both have 1 and -1 for Z_1 and X_1 respectively and should be grouped into the same whole plot. The same should be done for rows 2 and 3. You would need to do this grouping for the entire design.

In design 2, we see that there is an $\times$ in the eligible column. This indicates that there are no eligible assignments of Z_1 using the splitting the whole plot method for that particular design.

The next table is the catalogue of optimal designs for assigning Z_1 by the pseudo-partitioning method. These assignments do not require a re-arrangement of the whole plots after assigning the optimal Z_1.

The last table is the catalogue of optimal designs for assigning a split-plot factor z_1. This works the same as the table for splitting the whole plot. The exception being that you would append the optimal column to the right-hand side of the design and there is no need to re-arrange the whole plots.

Z_1	X_1	x_1	x_2
1	-1	-1	-1
-1	-1	1	-1
-1	-1	-1	1
1	-1	1	1
-1	1	-1	-1
1	1	1	-1
1	1	-1	1
-1	1	1	1
1	-1.73205	0	0
1	-1.73205	0	0
-1	-1.73205	0	0
-1	-1.73205	0	0
1	1.73205	0	0
1	1.73205	0	0
-1	1.73205	0	0
-1	1.73205	0	0
1	0	-1.73205	0
1	0	1.73205	0
-1	0	0	-1.73205
-1	0	0	1.73205
1	0	0	0
1	0	0	0
-1	0	0	0
-1	0	0	0

TABLE 3.21. Appending Z_1 from the optimal design table for design 1 in the splitting the whole plot method.

3.6.1 Unbalanced VKM Method CCD

Method: Unbalanced VKM Method CCD

Source: Equivalent Estimation Design Catalog

Design Number	Name	# WP Factors	# SP Factors	# WP's	*nrpw*	Runs
1	VKMU_CCD_D12	1	2	6	4	24
2	VKMU_CCD_D14	1	4	6	8	48
3	VKMU_CCD_D22	2	2	10	4	40
4	VKMU_CCD_D24	2	4	10	8	80
5	VKMU_CCD_D32	3	2	16	4	64
6	VKMU_CCD_D34	3	4	16	8	128

Splitting the Whole Plot Method

Design #	Eligible	# Eligible	# Optimal
1	✓	4	4
	1 -1 -1 1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
2	×		
3	✓	12	12
	1 -1 -1 1 1 -1 -1 1 -1 1 1 -1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
4	✓	216	128
	1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 -1 -1		
5	✓	131072	131072
	1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
6	*		

*Too many to search globally.

Pseudo-Partitioning Method

Design #	Eligible	# Eligible	# Optimal
1	×		
2	×		
3	×		
4	×		
5	×		
6	✓	2	2
	1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1		

Adding Split-Plot Method

Design #	Eligible	# Eligible	# Optimal
1	✓	4	4
	1 -1 -1 1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
2	×		
3	✓	20	20
	1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
4	✓	840	840
	1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1		
5	✓	131072	131072
	1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
6	*		

*Too many to search globally.

Method: Balanced VKM Method CCD

Source: Equivalent Estimation Design Catalog

Design Number	Name Factors	# WP Factors	# SP	# WP's	$nrpw$	Runs
7	VKM_CCD_D12	1	2	6	4	24
8	VKM_CCD_D13	1	3	8	8	64
9	VKM_CCD_D22	2	2	10	4	40
10	VKM_CCD_D23	2	3	12	4	48
11	VKM_CCD_D24	2	4	10	8	80
12	VKM_CCD_D32	3	2	16	4	64
13	VKM_CCD_D33	3	3	18	4	72
14	VKM_CCD_D34	3	4	16	8	128

Splitting the Whole Plot Method

<table>
<tr><th>Design #</th><th>Eligible</th><th># Eligible</th><th># Optimal</th></tr>
<tr><td rowspan="2">7</td><td>✓</td><td>4</td><td>4</td></tr>
<tr><td colspan="3">1 -1 -1 1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1</td></tr>
<tr><td>8</td><td colspan="3">Running</td></tr>
<tr><td rowspan="2">9</td><td>✓</td><td>12</td><td>12</td></tr>
<tr><td colspan="3">1 -1 -1 1 1 -1 -1 1 -1 1 1 -1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1
-1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1</td></tr>
<tr><td>10</td><td>×</td><td></td><td></td></tr>
<tr><td rowspan="2">11</td><td>✓</td><td>216</td><td>128</td></tr>
<tr><td colspan="3">1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 -1 -1 1
1 1 1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1
-1 1 1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 -1 -1</td></tr>
<tr><td rowspan="2">12</td><td>✓</td><td>131072</td><td>131072</td></tr>
<tr><td colspan="3">1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1
-1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1
-1 1 1 -1 -1 1 1 -1 -1</td></tr>
<tr><td>13</td><td>×</td><td></td><td></td></tr>
<tr><td>14</td><td>*</td><td></td><td></td></tr>
</table>

*Too many to search globally

Pseudo-Partitioning Method

<table>
<tr><th>Design #</th><th>Eligible</th><th># Eligible</th><th># Optimal</th></tr>
<tr><td>7</td><td>×</td><td></td><td></td></tr>
<tr><td>8</td><td>×</td><td></td><td></td></tr>
<tr><td>9</td><td>×</td><td></td><td></td></tr>
<tr><td>10</td><td>×</td><td></td><td></td></tr>
<tr><td>11</td><td>×</td><td></td><td></td></tr>
<tr><td>12</td><td>×</td><td></td><td></td></tr>
<tr><td rowspan="2">13</td><td>✓</td><td>2</td><td>2</td></tr>
<tr><td colspan="3">1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 1 1 1 1
-1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1</td></tr>
<tr><td rowspan="2">14</td><td>✓</td><td>2</td><td>2</td></tr>
<tr><td colspan="3">1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1
-1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1
-1 1 1 -1 -1 1 1 -1 -1</td></tr>
</table>

Adding Split-Plot Method

Design #	Eligible	# Eligible	# Optimal
7	✓	4	4
	1 -1 -1 1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
8	✓	466560	466560
	1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1		
9	✓	20	20
	1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
10	×		
11	✓	840	840
	1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1		
12	✓	131072	131072
	1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
13	✓	5760	5760
	1 1 -1 -1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
14	†		

† **V** exceeds memory storage

Method: Two-strata Rotatable Designs

Source: Wang, Vining, and Kowalski (2010)

Design Number	Name Factors	# WP Factors	# SP	# WP's	*nrpw*	Runs
15	WVK_2SR_D12	1	2	6	4	24
16	WVK_2SR_D13	1	3	8	8	64
17	WVK_2SR_D14	1	4	6	8	48
18	WVK_2SR_D21	2	1	10	2	20
19	WVK_2SR_D22	2	2	10	4	40
20	WVK_2SR_D23	2	3	12	8	96
21	WVK_2SR_D24	2	4	10	8	80
22	WVK_2SR_D31	3	1	16	2	32
23	WVK_2SR_D32	3	2	16	4	64
24	WVK_2SR_D33	3	3	18	8	144
25	WVK_2SR_D34	3	4	16	8	128

Splitting the Whole Plot Method

Design #	Eligible	# Optimal	# Eligible
15	✓	4	4
	1 -1 -1 1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
16	×		
17	×		
18	×		
19	✓	12	12
	1 -1 -1 1 1 -1 -1 1 -1 1 1 -1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
20	✓	25754112	25754112
	1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1		
21	✓	216	144
	1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 -1 -1		
22	×		
23	✓	140	140
	1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
24	*		
25	*		

*Too many to search globally

Pseudo-Partitioning Method

Design #	Eligible	# Optimal	# Eligible
15	×		
16	×		
17	×		
18	×		
19	×		
20	×		
21	×		
22	✓	2	2
	1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1		
23	✓	2	2
	1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1		
24	✓	2	2
	1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1		
25	✓	2	2
	1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1		

Adding Split-Plot Method

Design #	Eligible	# Optimal	# Eligible
15	✓	4	4
	1 -1 -1 1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
16	✓	640	640
	1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
17	×		
18	×		
19	✓	20	20
	1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
20	running		
21	✓	840	840
	1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1		
22	×		
23	✓	844	844
	1 1 -1 -1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
24	*		
25	*		

*Too many to search globally

Method: CUBE Design

Source: Draper and John (1998)

Design Number	Name Factors	# WP Factors	# SP	# WP's	*nrpw*	Runs
26	CUBE_D12	1	2	4	4	16
27	CUBE_D22	2	2	6	4	24
28	CUBE_D32	3	2	10	4	40
29	CUBE_D42	4	2	18	4	72

Splitting the Whole Plot Method

Design #	Eligible	# Optimal	# Eligible
26	✓	4	4
	1 -1 -1 1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1		
27	✓	12	12
	1 -1 -1 1 1 -1 -1 1 -1 1 1 -1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1		
28	✓	140	140
	1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1		
29	*		

*Too many to search globally.

Pseudo-Partitioning Method

Design #	Eligible	# Optimal	# Eligible
26	×		
27	×		
28	✓	2	2
	1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1		
29	*		

*Too many to search globally

Adding Split-Plot Method

Design #	Eligible	# Optimal	# Eligible
26	✓	4	4
	1 -1 -1 1 -1 1 1 -1 1 1 -1 -1 1 1 -1 -1		
27	✓	20	20
	1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
28	✓	844	844
	1 1 -1 -1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
29	*		

*Too many to search globally

Method: Balanced VKM Method SCD

Source: Equivalent Estimation Catalogue

Design Number	Name Factors	# WP Factors	# SP	# WP's	*nrpw*	Runs
30	VKM_SCD_D12	1	2	2	4	24
31	VKM_SCD_D13	1	3	6	12	72
32	VKM_SCD_D22	2	2	10	4	40
33	VKM_SCD_D24	2	4	4	8	80
34	VKM_SCD_D32	3	2	16	4	64

Splitting the Whole Plot Method

Design #	Eligible	# Eligible	# Optimal
30	✓	32	32
	1 -1 1 -1 1 -1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1		
31	†		
32	×		
33	✓	393216	253952
	1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 1 1 -1 -1 -1 -1		
34	×		

† **V** exceeds memory storage

Pseudo-Partitioning Method

Design #	Eligible	# Optimal	# Eligible
30	×		
31	×		
32	×		
33	×		
34	×		

Adding Split-Plot Method

<table>
<tr><th>Design #</th><th>Eligible</th><th># Optimal</th><th># Eligible</th></tr>
<tr><td rowspan="2">30</td><td>✓</td><td>32</td><td>32</td></tr>
<tr><td colspan="3">1 -1 1 -1 1 -1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1</td></tr>
<tr><td>31</td><td>†</td><td></td><td></td></tr>
<tr><td rowspan="2">32</td><td>✓</td><td>512</td><td>512</td></tr>
<tr><td colspan="3">1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1
-1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1</td></tr>
<tr><td rowspan="2">33</td><td>✓</td><td>393216</td><td>393216</td></tr>
<tr><td colspan="3">1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1
-1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1
-1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1</td></tr>
<tr><td>34</td><td>×</td><td></td><td></td></tr>
</table>

† **V** exceeds memory storage

3.7 An Example: A Return to the Ceramic Pipe

In the introduction we looked at the example of a split-plot experiment from Vining and Kowalski (2008) where the goal was to find the optimal settings to maximize the strength of a ceramic pipe. There were two hard-to-vary factors (temperature in furnace zone 1 and temperature in furnace zone 2) and two easy-to-vary factors (amount of binder in batch and grinding speed). The design with the corresponding response variable for each run is found in Table 1.2.

We suggested that, in a different version of this experiment, there may have been an added categorical factor of interest. For a hypothetical whole-plot factor, this was taken to be the type of furnace and for the split-plot factor this was taken to be the placement of the pipe. As we already have a base design in the quantitative factors (that was actually used in an experiment because it was deemed "good"), we can run this design through the algorithm to find D-optimal designs for adding a whole-plot factor and a split-plot factor respectively.

When running the algorithm for adding the split-plot factor, there are 10 eligible assignments of the factorial points. Using those and then appending on possible assignments to the axial points, there are 20 eligible assignments for the entire design. All of those 20 assignments are optimal. The D-optimal assignment of z_1 that is given by the algorithm is in the Table 3.22. It so happens that this assignment is equivalent to $z_1 = -X_1X_2x_2$ in the factorial points.

When running the algorithm for adding the whole-plot factor, we find that it fails to find any eligible assignments under the pseudo-partitioning method but there are potential assignments using the splitting the whole plot method. The algorithm shows that using the splitting the whole plot method, there are 6 eligible assignments of the categorical variable Z_1 to the factorial whole plots. Using those as a starting point and then adding in assignments for the axial whole plots, reveals that there are 12 eligible assignments of Z_1 to the design. All of those assignments are D-optimal. Appending the algorithm's suggested assignment of Z_1 to to the design in Table 1.2 is shown in Table 3.23. To make use of this design, you would then need to resort it back into whole plots as in Table 3.24.

This shows in practice what a practitioner would do. They would take a good design from the literature for the quantitative factors of their experiment and then they would apply the algorithm to get the assignment of the categorical factor to append to their design. To run this experiment, they would randomly select a whole plot in the new design and, within that whole plot, randomly select runs to conduct in random order. This would be repeated until all runs were finished.

Originally, we know the ceramic pipe example was run in just the quantitative factors with the observed responses in Table 1.2. Vining and Kowalski used the output to fit a full second-order model of the form:

$$E(Y) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 x_1 + \beta_4 x_4 + \beta_{11} X_1^2 + \beta_{22} X_2^2 + \beta_{33} x_1^2 + \beta_{44} x_2^2$$
$$+ \beta_{12} X_1 X_2 + \beta_{13} X_1 x_1 + \beta_{14} X_1 x_2 + \beta 23 X_2 x_3 + \beta_{24} X_2 x_2 + \beta_{34} x_1 x_2.$$

The estimated coefficients for this fitted model can be found in Table 3.25.

Suppose instead, that the observed responses were the output from a design where we had added on the split-plot categorical factor, $z_1 = 1$, the positioning of

X_1	X_2	x_1	x_2	z_1	X_1	X_2	x_1	x_2	z_1
-1	-1	-1	-1	1	0	-1	0	0	1
		1	-1	1			0	0	1
		-1	1	-1			0	0	-1
		1	1	-1			0	0	-1
1	-1	-1	-1	-1	0	1	0	0	1
		1	-1	-1			0	0	1
		-1	1	1			0	0	-1
		1	1	1			0	0	-1
-1	1	-1	-1	-1	0	0	-1	0	1
		1	-1	-1			1	0	1
		-1	1	1			0	-1	-1
		1	1	1			0	1	-1
1	1	-1	-1	1	0	0	0	0	1
		1	-1	1			0	0	1
		-1	1	-1			0	0	-1
		1	1	-1			0	0	-1
-1	0	0	0	1	0	0	0	0	1
		0	0	1			0	0	1
		0	0	-1			0	0	-1
		0	0	-1			0	0	-1
1	0	0	0	1	0	0	0	0	1
		0	0	1			0	0	1
		0	0	-1			0	0	-1
		0	0	-1			0	0	-1

TABLE 3.22. The suggested optimal designs for the hypothetical ceramic pipe experiment with an added categorical split-plot factor.

Z_1	X_1	X_2	x_1	x_2	Z_1	X_1	X_2	x_1	x_2
1	-1	-1	-1	-1	1	0	-1	0	0
-1	-1	-1	1	-1	1	0	-1	0	0
-1	-1	-1	-1	1	-1	0	-1	0	0
1	-1	-1	1	1	-1	0	-1	0	0
1	1	-1	-1	-1	1	0	1	0	0
-1	1	-1	1	-1	1	0	1	0	0
-1	1	-1	-1	1	-1	0	1	0	0
1	1	-1	1	1	-1	0	1	0	0
-1	-1	1	-1	-1	1	0	0	-1	0
1	-1	1	1	-1	1	0	0	1	0
1	-1	1	-1	1	-1	0	0	0	-1
-1	-1	1	1	1	-1	0	0	0	1
-1	1	1	-1	-1	1	0	0	0	0
1	1	1	1	-1	1	0	0	0	0
1	1	1	-1	1	-1	0	0	0	0
-1	1	1	1	1	-1	0	0	0	0
1	-1	0	0	0	1	0	0	0	0
1	-1	0	0	0	1	0	0	0	0
-1	-1	0	0	0	-1	0	0	0	0
-1	-1	0	0	0	-1	0	0	0	0
1	1	0	0	0	1	0	0	0	0
1	1	0	0	0	1	0	0	0	0
-1	1	0	0	0	-1	0	0	0	0
-1	1	0	0	0	-1	0	0	0	0

TABLE 3.23. Appending the suggested optimal assignment of Z_1 to the base design for the hypothetical ceramic pipe experiment with an added categorical whole-plot without resorting into whole plots.

Z_1	X_1	X_2	x_1	x_2	Z_1	X_1	X_2	x_1	x_2
1	-1	-1	-1	-1	1	0	-1	0	0
			1	1				0	0
-1	-1	-1	-1	1	-1	0	-1	0	0
			1	-1				0	0
1	1	-1	-1	-1	1	0	1	0	0
			1	1				0	0
-1	1	-1	-1	1	-1	0	1	0	0
			1	-1				0	0
-1	-1	1	-1	-1	1	0	0	-1	0
			1	1				1	0
1	-1	1	-1	1	-1	0	0	0	-1
			1	-1				0	1
-1	1	1	-1	-1	1	0	0	0	0
			1	1				0	0
1	1	1	-1	1	-1	0	0	0	0
			1	-1				0	0
1	-1	0	0	0	1	0	0	0	0
			0	0				0	0
-1	-1	0	0	0	-1	0	0	0	0
			0	0				0	0
1	1	0	0	0	1	0	0	0	0
			0	0				0	0
-1	1	0	0	0	-1	0	0	0	0
			0	0				0	0

TABLE 3.24. The suggested optimal designs for the hypothetical ceramic pipe experiment with an added categorical whole-plot factor after resorting the runs into whole plots.

Term	Estimated Coefficient
Intercept	74.9055
X_1	4.5579
X_2	-6.5592
X_1^2	1.7381
X_2^2	-0.5407
$X_1 X_2$	0.8431
x_1	-4.9730
x_2	4.0922
x_1^2	-2.3864
x_2^2	2.5736
$x_1 x_2$	-1.0394
$X_1 x_1$	1.4356
$X_1 x_2$	-1.4794
$X_2 x_1$	-1.0019
$X_2 x_2$	1.9856

TABLE 3.25. Estimated coefficients for the ceramic pipe experiment in the quantitative factors as given in Vining and Kowalski (2008)

the pipe in the furnace. Note, there was no categorical factor when the data was actually obtained. This is an artificial example to show how we would fit the model if this were output collected with the categorical variable present. Let us see how we would instead fit our model and look at the coefficients in this instance.

Recall that our goal is to fit a design that has common second-order and interaction terms but has separate model terms for the intercept and first-order terms at each level of z_1. The model we would like to fit is given by:

$$E(Y) = \sum_{z_1} W_{z_1} \left(\beta_0(z_1) + \beta_1(z_1)X_1 + \beta_2(z_1)X_2 + \beta_3(z_1)x_1 + \beta_4(z_1)x_2\right)$$

$$+ \sum_{i=1}^{2} \beta_{ii} X_i^2 + \sum_{i=3}^{4} \beta_{ii} x_i^2 + \beta_{12}X_1X_2 + \beta_{34}x_1x_2 + \sum_{i=1}^{2}\sum_{j=3}^{4} X_i x_j. \qquad (3.5)$$

How will we do this? As the addition of z_1 does not alter the conditions needed for OLS-GLS equivalence (whole-plot are balanced, subplots are orthogonal, all split-plot axial points are contained in the same whole-plot), we can estimate the coefficients with ordinary least-squares. However, we must first take care to properly make our design matrix.

Recall in Chapter 2 that we were writing our model for the design with separate estimates of the coefficients for each first-order term in equation 3.5. In the design matrix, we needed two intercept terms $I(-1)$ and $I(1)$ and we needed two columns for each first order term. For example, to estimate the coefficients for x_1 we would need a column $x_1(-1)$, which takes on the value of x_1 when $z_1 = -1$ and equals 0 when $z_1 = 1$ as well as a column $x_1(1)$ which takes on the value of x_1 when $z_1 = 1$ and equals 0 when $z_1 = -1$. All second order designs are estimated with a single column. Applying that approach to fit the model in equation 3.5, the design matrix is given by:

	Ip	Im	X1p	X1m	X2p	X2m	x1p	x1m	x2p	x2m	X1X1	X2X2	X1X2	x1x1	x2x2	x1x2	X1x1	X1x2	X2x1	X2x2
1	1	0	-1	0	-1	0	-1	0	-1	0	1	1	1	1	1	1	1	1	1	1
2	1	0	-1	0	-1	0	1	0	-1	0	1	1	1	1	1	-1	-1	1	-1	1
3	0	1	0	-1	0	-1	0	-1	0	1	1	1	1	1	1	-1	1	-1	1	-1
4	0	1	0	-1	0	-1	0	1	0	1	1	1	1	1	1	1	-1	-1	-1	-1
5	0	1	0	1	0	-1	0	-1	0	-1	1	1	-1	1	1	1	-1	-1	1	1
6	0	1	0	1	0	-1	0	1	0	-1	1	1	-1	1	1	-1	1	-1	-1	1
7	1	0	1	0	-1	0	-1	0	1	0	1	1	-1	1	1	-1	-1	1	1	-1
8	1	0	1	0	-1	0	1	0	1	0	1	1	-1	1	1	1	1	1	-1	-1
9	0	1	0	-1	0	1	0	-1	0	-1	1	1	-1	1	1	1	1	1	-1	-1
10	0	1	0	-1	0	1	0	1	0	-1	1	1	-1	1	1	-1	-1	1	1	-1
11	1	0	-1	0	1	0	-1	0	1	0	1	1	-1	1	1	-1	1	-1	-1	1
12	1	0	-1	0	1	0	1	0	1	0	1	1	-1	1	1	1	-1	-1	1	1
13	1	0	1	0	1	0	-1	0	-1	0	1	1	1	1	1	1	-1	-1	-1	-1
14	1	0	1	0	1	0	1	0	-1	0	1	1	1	1	1	-1	1	-1	1	-1
15	0	1	0	1	0	1	0	-1	0	1	1	1	1	1	1	-1	-1	1	-1	1
16	0	1	0	1	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1
17	1	0	-1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
18	1	0	-1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
19	0	1	0	-1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
20	0	1	0	-1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
21	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
22	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
23	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
24	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
25	1	0	0	0	-1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
26	1	0	0	0	-1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
27	0	1	0	0	0	-1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
28	0	1	0	0	0	-1	0	0	0	0	0	1	0	0	0	0	0	0	0	0

29	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
30	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
31	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
32	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
33	1	0	0	0	0	0	-1	0	0	0	0	0	0	1	0	0	0	0	0	0
34	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
35	0	1	0	0	0	0	0	0	0	-1	0	0	0	0	1	0	0	0	0	0
36	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
37	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
42	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
43	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
46	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
47	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
48	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Using this as our design matrix $\mathbf{X}$, the OLS estimates of our fitted coefficients is derived from the equation $\hat{\beta} = (X'X)^{-1} X'y$. Our new estimated coefficients are given in Table 3.26.

To help visualize the response surface, as in Chapter 1, we will make contour plots for both $z_1 = 1$ and $z_1 = -1$. However, unlike before, we now have four quantitative factors and we cannot simply make a single contour plot for each level of z_1. Instead, as we have four combinations of whole-plot factor levels that each have two split-plot factors that vary within them, we will make four contour plots for each level of z_1. Each combination of whole-plot factor levels will get its own

Model Term	Estimated Coefficient
$I(1)$	74.9103788
$I(-1)$	-74.8581061
$X_1(1)$	4.6954167
$X_1(-1)$	-4.4204167
$X_2(1)$	-6.7504167
$X_2(-1)$	6.3679167
$x_1(1)$	-4.9290000
$x_1(-1)$	5.0287500
$x_2(1)$	4.2512500
$x_2(-1)$	-3.9650000
X_1X_1	1.7532197
X_2X_2	-0.5255303
X_1X_2	0.9862500
x_1x_1	-2.4155303
x_2x_2	2.5967424
x_1x_2	-1.0393750
X_1x_1	1.4356250
X_1x_2	-1.6706250
X_2x_1	-1.0018750
X_2x_2	2.1231250

TABLE 3.26. The fitted coefficients for the ceramic pipe example including categorical split-plot factor pipe position with no adjustments to the response variables.

contour plot. Not only will this allow us to see if there are different solutions for each level of z_1 but we can also compare solutions between whole plots.

While we could select any two factors to use to make the four contour plots, it makes sense to fix the whole-plot combinations. As there is more precision in the estimating the split-plot factors, it makes sense to first examine the contour plots that will map changes for the split-plot factors within the whole plots.

We can see the contour plots for $z_1 = 1$ in Figure 3.1 and the contour plots for $z_1 = -1$ in Figure 3.2. In each of these figure from top to bottom and left to right, the contour plots are for the whole-plot factor combinations (X_1, X_2) of $(1, 1)$, $(1, -1)$, $(1, -1)$, and $(-1, -1)$ respectively.

Though this is an artificial assignment of which responses had $z_1 = -1$ and $z_1 = 1$ after the fact, we can look at the contour plots to get a sense of the information they give us. Firstly, we see that in our design space $[-1, 1] \times [1, 1]$, the maximum in all of the plots is around the point $x_1 = 1$ and $x_2 = -1$ regardless of the level of z_1 or the particular whole plot. If we are looking merely for the optimizing value, our level of z_1 may not be of particular importance, which is understandable because of the "arbitrary" addition of z_1 in the example. We also see that the center of the contour plots for all levels of z_1 form a minimum. This means our local maximum will be on a boundary of the design space as we saw.

Let us now look at how a shift in our response variable at different values of z_1 would affect our response surface and contour plots. One of our goals of RSM might be to identify the literal optimal value on our design space. This is affected by vertical shifts in the response surface. Suppose we were to add a constant amount

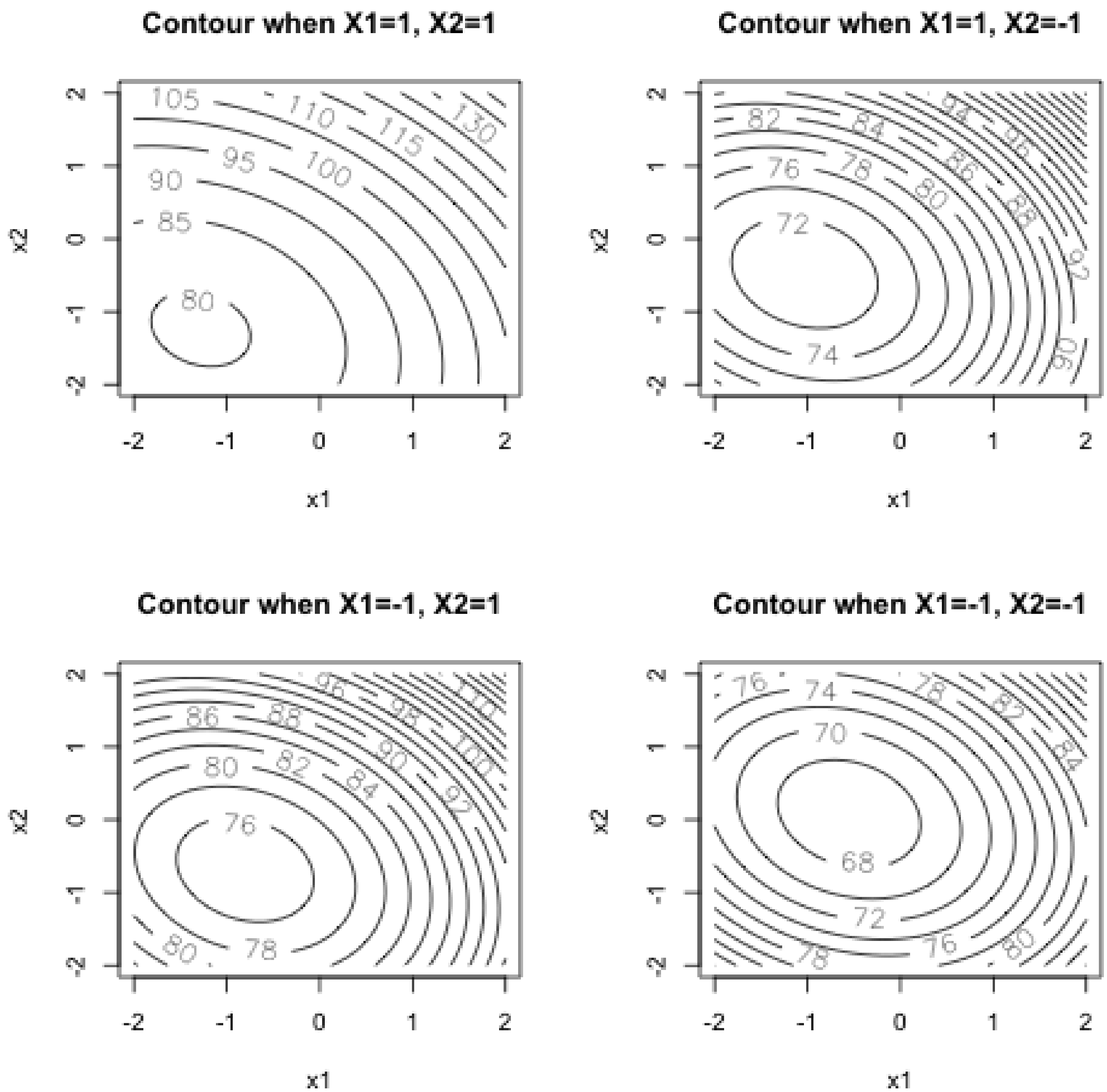

FIGURE 3.1. The contour plots of the response surface at all four combinations of the whole-plot factors in the ceramic pipe example when $z_1 = -1$.

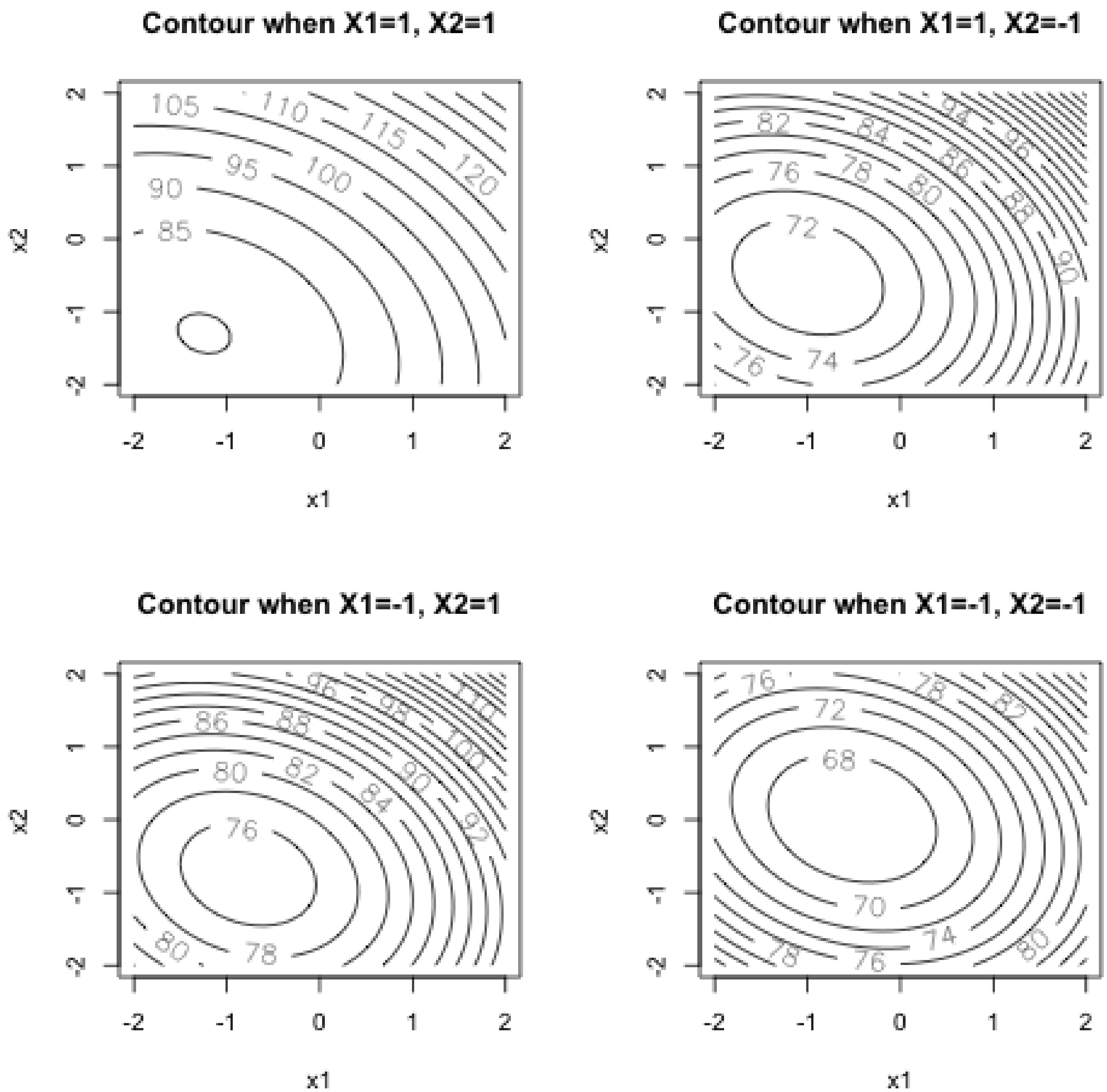

FIGURE 3.2. The contour plots of the response surface at all four combinations of the whole-plot factors in the ceramic pipe example when $z_1 = 1$.

to each the response at each run where $z_1 = 1$. We should see that response surface shift upwards by 10 units in the contour but the shape should remain the same.

To show this, we will refit the coefficients for the model in equation 3.5 using the new y-values. These are given in Table 3.27.

The contour plots for the new vertically shifted model can be seen in Figures 3.3 and 3.4 for $z_1 = -1$ and $z_1 = 1$ respectively.

By comparing these contours to the unshifted contours in figures 3.1 and 3.2 we can clearly see how the additive effect in $z_1 = 1$ affects the contours. The contours where $z_1 = -1$ remain identical but when we look at the contours for $z_1 = 1$, we notice the shift. While the shape is the same, the levels on the contours have all increased by 10. Vertical shifts do not affect the shape, only the height. Were we to get similar contours at $z_1 = 1$ and $z_1 = -1$ except at the values on the levels of the contour plot, we would know that it was a vertical shift we were observing. Now let us see how a different type of shift would affect the response curves.

In Chapter 2, we learned that it was the coefficients in front of the first order term that controlled the location when we had common second order terms. To simulate this, we will add a factor of 10 to the response value y for each run where we meet both of two conditions: $x_1 = 1$ (so that the level of x_1 has an effect on the location) and where $z_1 = 1$ (so that it only happens in one of the response surfaces). Note that this will also affect the shape of the response curve because there will be an interaction effect and not just an additive effect.

Using ordinary least-squares to fit the model with the new y-values gives the coefficients in Table 3.28.

Written in the style of equation 2.13 our fitted model is:

Model	Estimated Coefficient
$I(1)$	76.1225000
$(I(-1)$	74.2520455
$X_1(1)$	4.6954167
$X_1(-1)$	4.4204167
$X_2(1)$	-6.7504167
$X_2(-1)$	-6.3679167
$x_1(1)$	0.07100000
$x_1(-1)$	-5.0287500
$x_2(1)$	4.2512500
$x_2(-1)$	3.9650000
X_1X_1	1.2986742
X_2X_2	-0.9800758
X_1X_2	0.9862500
x_1x_1	0.7662879
x_2x_2	2.5967424
x_1x_2	-1.0393750
X_1x1	1.4356250
X_1x_2	-1.6706250
X_2x_1	-1.0018750
X_2x_2	2.1231250

TABLE 3.27. The fitted coefficients for the ceramic pipe example when there is an addition of 10 to the response variable at all runs where $z_1 = 1$.

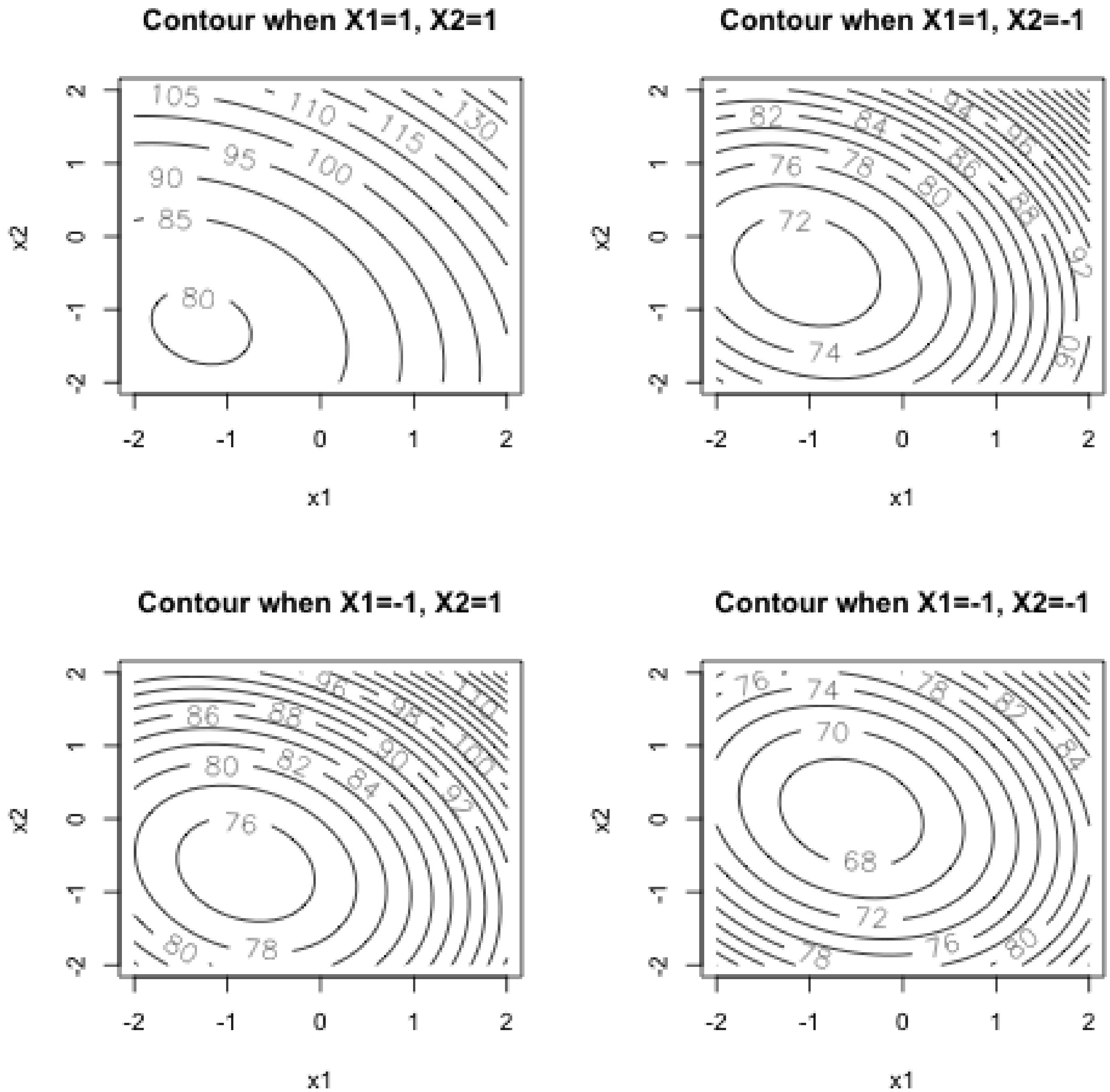

FIGURE 3.3. The contour plots of the response surface at all four combinations of the whole-plot factors in the ceramic pipe example when $z_1 = -1$ and we add a constant amount to all responses when $z_1 = 1$.

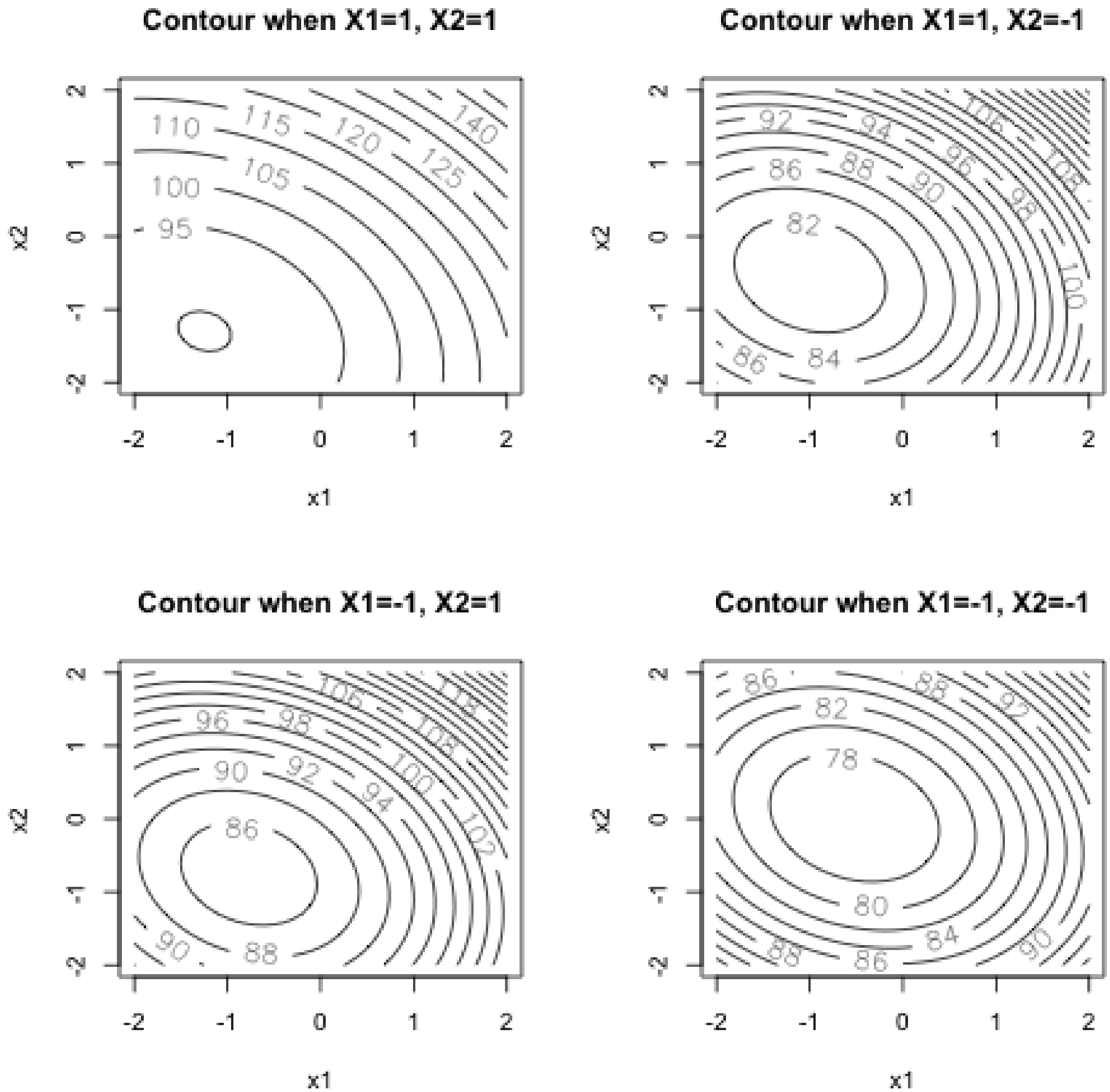

FIGURE 3.4. The contour plots of the response surface at all four combinations of the whole-plot factors in the ceramic pipe example when $z_1 = 1$ and we add a constant amount to all responses when $z_1 = 1$.

Model Term	Estimated Coefficient
$I(1)$	84.9103788
$I(-1)$	74.8581061
$X_1(1)$	4.6954167
$X_1(-1)$	4.4204167
$X_2(1)$	-6.7504167
$X_2(-1)$	-6.3679167
$x_1(1)$	-4.9290000
$x_1(-1)$	-5.0287500
$x_2(1)$	4.2512500
$x_2(-1)$	3.9650000
X_1X_1	1.7532197
X_2X_2	-0.5255303
X_1X_2	0.9862500
x_1x_1	-2.4155303
x_2x_2	2.5967424
x_1x_2	-1.0393750
X_1x_1	1.4356250
X_1x_2	-1.6706250
X_2x_1	-1.0018750
X_2x_2	2.1231250

TABLE 3.28. The fitted coefficients for the ceramic pipe example when there is an addition of 10 to the response variable at all runs where $z_1 = 1$ and $x_1 = 1$.

$$
\begin{aligned}
\hat{y} = & W_1(84.9104 + 4.6954X_1 - 6.7504X_2 - 4.9290x_1 + 4.2513x_2) \\
& + W_{-1}(74.8581 - 4.4204X_1 - 6.3679X_2 - 5.0288x_1 + 3.9650x_2) \\
& + 1.7532X_1^2 - 0.5255X_2^2 - 2.4155x_1^2 + 2.5967x_2^2 \\
& - 1.0393x_1x_2 + 1.4356X_1x_1 - 1.6706X_1x_2 - 1.0019X_2x_1 + 2.1231X_2x_2. \quad (3.6)
\end{aligned}
$$

Now we can make contour plots of the fitted model to see if we observe the expected difference in the response surface.

In Figure 3.5, we can see four contour plots for when $z_1 = -1$ and in Figure 3.6, we can see the four contour plots for when $z_1 = 1$.

Comparing these two contour plots we can see two distinct features. In the contour plots for $z_1 = 1$, the center of the graph has shifted to the left towards $x_1 = -1$. Note that due to a shape change, this center is neither a maximum nor a minimum but a saddle point. As our center was previously a minimum and we are adding on to y when $x_1 = 1$, it makes sense that this center would move away from $x_1 = 1$.

By separating out the different response surfaces, we are able to see two very clear differences in our response for both levels of the categorical variable that would not have been as clearly visible had we graphed z_1 like it was a quantitative variable. This motivates both why we need the separate response surfaces and that our fitted model is able to capture these differences.

In practice, after fitting the data and doing our initial data exploration with contour plots, we would follow standard procedures in the analysis of arising from

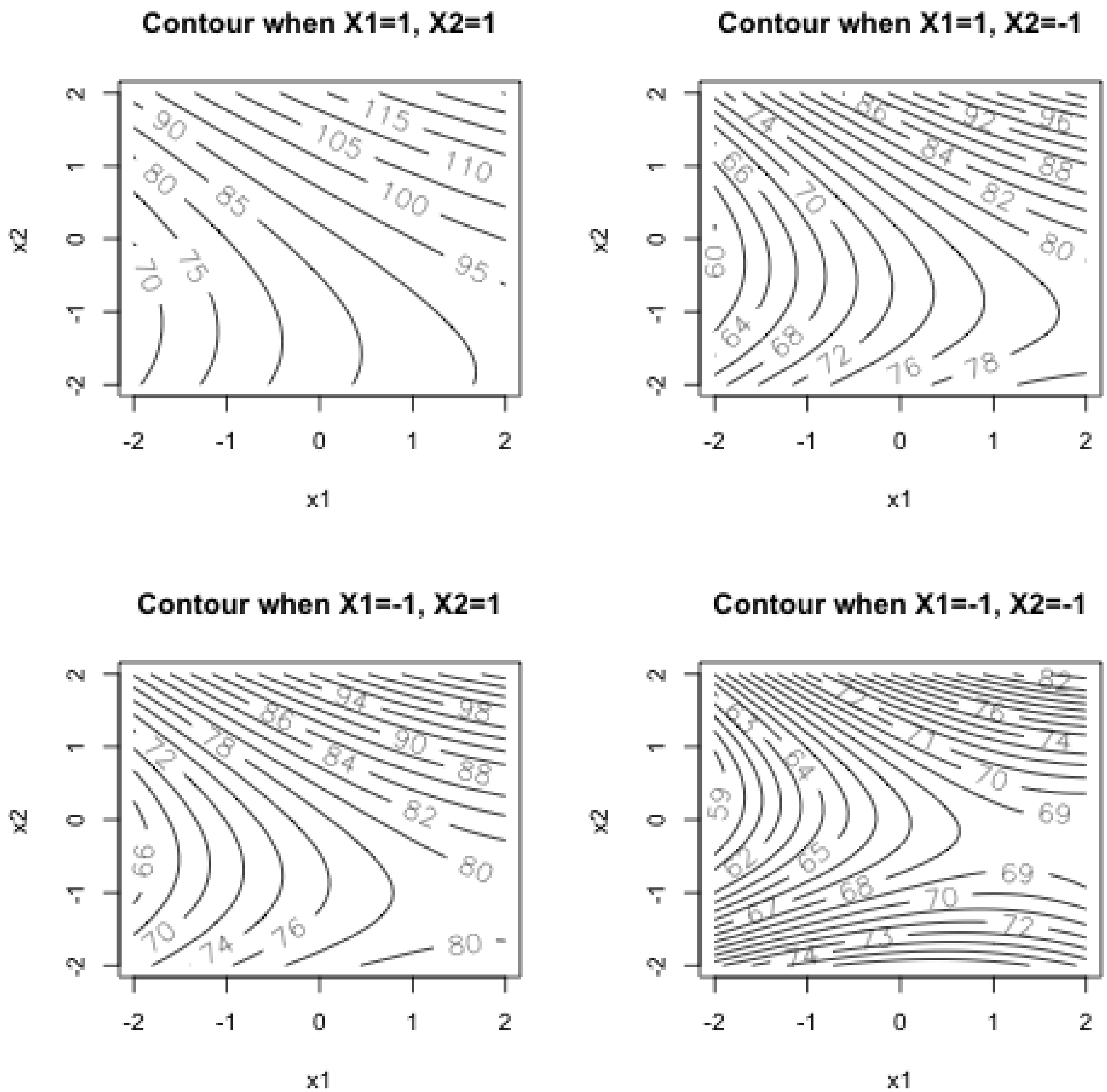

FIGURE 3.5. The contour plots of the response surface at all four combinations of the whole-plot factors in the ceramic pipe example when $z_1 = -1$.

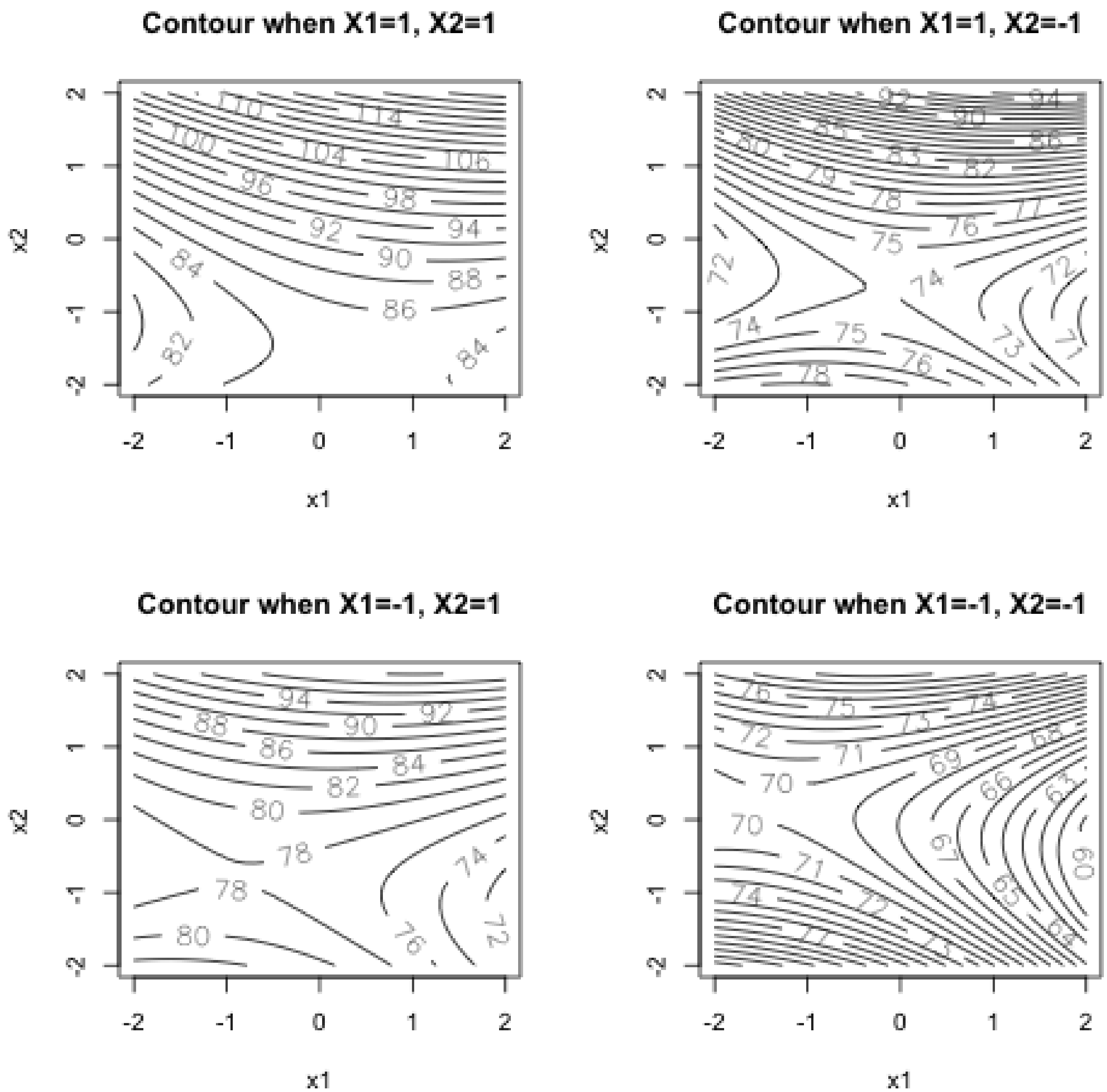

FIGURE 3.6. The contour plots of the response surface at all four combinations of the whole-plot factors in the ceramic pipe example when $z_1 = 1$.

designed experiments, using mixed models, because of the split-plot structure. As there are many terms in our model, we would first find estimates of our variances using methods such as restricted maximum likelihood (REML). We could be particularly interested in seeing whether certain groups of terms could be eliminated from the model. We might be able to eliminate one or more of the factors, or some of the interactions. We would be particularly interested in establishing whether the categorical factor is even necessary, as we purposefully chose initial base designs that were good in the quantitative factors in case the categorical factors was not necessary from a statistical significance point of view. In all of these cases, we would test the full versus the proposed reduced model to try to eliminate unnecessary terms.

In choosing our model, we have also made several assumptions on the relationship between our factors which we should examine using goodness-of-fit tests to make sure we have not misspecified our model. The large assumption we made is that there are common second-order terms for both levels of the categorical factor. If we have enough points to estimate separate coefficients for the second-order terms at each level of z_1 we can formally test whether the coefficients are equal. This would be our ideal scenario. However, as our runs are at a premium, this may not be possible if we do not have a sufficient number of degrees of freedom to allocate to those model terms. In this case, we may be able to check for goodness of fit of this second-order model by testing in a way that is analogous to the 1 degree of freedom test to test for curvature in an unreplicated 2-level factorial design with center points.

Finally, we must remember what our goals for RSM are. If we want to know if both levels of the categorical factor should use the same levels of the quantitative factors in practice, we need to test whether the response surfaces have a common optimizing value, so could test whether the qualitative factor has a simple additive effect (if it has an effect at all).

3.8 Summary

In this chapter, we looked at four different methods for creating a response surface split-plot design with a categorical factor. All of these methods relied on taking a base design for the quantitative factors from the literature for that was already considered to have good properties and then adding on the categorical factor second. There were three methods for adding a whole-plot factor (doubling the design, splitting the whole plot, pseudo-replication) and one for adding a split-plot factor. We discussed in detail how each of these methods worked and explored the conditions under which an assignment of the categorical factor would be deemed feasible.

All of these methods were coded into an algorithm that was implemented in R to produce tables of D-optimal designs for adding a categorical factor to published tables of designs in the quantitative split-plot RSM literature. We saw that not all methods were appropriate for every base design and we were some times limited in which method could be used.

We followed this up with an example of a split-plot RSM design that had a categorical factor added on. We used the algorithm to find a D-optimal assignment of a categorical split-plot factor and then fitted a second-order model that had separate coefficients for the first-order terms at each level of the categorical factor. This induced separate response surfaces for each level of z_1 and we explored the differences in the response surfaces by using contour plots.

This research allowed us to bridge the gap between RSM for categorical factors and RSM for split-plot designs and provide a foundation for the correct way to add categorical factors when there are randomization restrictions in play. Apart from contributing tables of optimal designs, this also established results on how to use appropriately use generating factors to produce feasible assignments of the categorical factors.

Chapter 4

Future Work & Conclusions

4.1 Further Analysis and Tests

At the end of chapter 3, we went through some standard analysis tools that experimenters would use after conducting an experiment to answer our questions about the response surface, the optimal value, and the optimizing value. There are a few aspects of these analysis issues that could be different for our RSM split-plot designs with categorical factors that deserve some further attention.

At the end of chapter 3 we spoke of using standard approaches for testing lack-of-fit and for common coefficients. These tests will suffice though there is merit to future research on determining whether there is a difference in the variance-covariance for the coefficients that are specific to certain levels of the categorical factor. The columns for these model terms in the design matrix have 0's in half the entries for when we are not at the appropriate level of the categorical variable. Even though, for example, four runs may come from the same whole plot, within the whole plots half of the entries for the whole-plot factors will be 0 in the design matrix accordingly.

Research into how this affects the variance estimates is of importance to see if it will be different than the variance terms for testing the second-order factor effects and interaction effects.

Another issue surrounds estimating the optimizing value. At the end of section 2.1, we detailed how we could write out the response surface for two quantitative factors as either a linear model as in equation 2.19 or as a non-linear vertex model as is equation 2.20. While the regression model had the ease of being estimated more easily as a linear model, the vertex model had coefficients that had more meaningful interpretations. For the two quantitative factor scenario, we were able to find an algebraic connection between the two sets of coefficients to show that not all terms in the regression model were equally as important for testing for the optimizing value. In all of our designs that we use in this thesis, we have more than two quantitative factors. A next step in the research is to see how much easier it is to isolate the important variables for testing the optimizing value and decide the appropriate nonlinear test for the optimizing value of our response surface for a given level of z_1 or Z_1.

On a final note, our ceramic pipe example also made use of an OLS-GLS equivalent designs for fitting our coefficients. We should note that different tools are necessary for the base designs where we do not meet OLS-GLS equivalence conditions or for instances where our addition of the categorical factor causes the design to lose the OLS-GLS equivalence property. This means that several different analysis tools will be needed to fit and assess the fit of our models depending on the base design and potentially based on the assignment method. Whether the assignment method of splitting the whole-plot or pseudo-partitioning makes a difference in whether or not we have OLS-GLS equivalence or which technique needs to be used should be examined as well.

4.2 Extensions of the Algorithm

Now that we have established areas of future research on the analysis to test for differences in the response surfaces and for lack of fit in our models, we need to go back a bit to look at the algorithm itself. In chapter 3, we discussed some of the limitations on the base designs that could be entered into the algorithm in the interests of thoroughly exploring the most common factorial design setting of balanced designs with a single factor.

Having fully explored these design types, a natural extension for future research is to modify the algorithm to accommodate unbalanced designs and designs with more than one categorical factor or more than two levels to the categorical factor.

4.2.1 Multiple Levels and Multiple Categorical Factors

When adding a categorical factor for response surface methodology, we discussed early on in the thesis how it was a somewhat naive approach to just treat the categorical factor as any other factor when making assignments. There are several reasons including not having levels appropriate for axial points but the main reason was that there is no interpretation to interpolated values on a response surface for a categorical factor. The point is that we wanted a separate response surface at each level of the categorical factor and it was important that, in the subsetted designs (subsetted by level of the categorical factor), that we were able to estimate a complete response surface.

Though we may not have a sense of ordering a categorical factor with more than two levels (i.e. assigning levels -1, 0, 1 may be completely arbitrary), it is certainly possible that a categorical factor could have more than two levels. This will have a different effect on our method depending on which way we are adding the categorical factor.

If we are splitting the whole plot, we would require that the number of rows per whole plot be a multiple of the number of levels of the categorical factor and be at least twice as large as the number of levels to ensure we are not creating whole plots of size one when we redivide the whole plots. For three levels of the categorical factor, this means we require whole plots of at least size 6 and that are multiples of threes. This presents the challenge of both needing large whole plots and designs that are not based on two-level factorial designs which are much less common in the literature.

If we are pseudo-partitioning, instead of doubling the number of axial whole plots, we would need to create even more new whole plots. For three levels of the categorical factor, it would require three copies of each axial whole plot which could be extremely large and run size inefficient.

We will also have the concern of needing enough factorial points at each level of the categorical factor to clearly estimate the terms in a first-order model. As there are more levels, we will need more points in general.

Thus, this is not to say that we cannot accommodate more levels, but we may need to be willing to deal with larger designs or look into three-level designs depending on the need.

The issue that needs to be worked around for accommodating multiple categorical factors is, in fact, very much the same because of the way it is commonly treated in the literature.

For two categorical factors, supposing they have two levels each, there are in essence $2^2 = 4$ different combinations of settings of the categorical factors. Each of the combinations has the potential to create a separate response surface. Thus we realize, our goal is not so much to fit a response surface at each level of the categorical factors as in the single factor setting, but rather to fit a response curve at each combination of categorical factors.

This would suggest that the appropriate method for handling the categorical factors is to create a "super factor" of sort that acts as an indicator of which combination of categorical factors are being used. For example, if there are two categorical factors z_1 and z_2 with two levels (-1 and +1) each, then we could create the super factor z given by:

$$z = \begin{cases} 1 & \text{if } z_1 = 1, z_2 = 1 \\ 2 & \text{if } z_1 = 1, z_2 = -1 \\ 3 & \text{if } z_1 = -1, z_2 = 1 \\ 4 & \text{if } z_1 = -1, z_2 = -1 \end{cases}$$

This factor acts as an identifier of which of the four response surfaces the run would contribute to in the subsetted designs. Clearly we would need to continue with the conditions that: we have enough points to clearly estimate a complete first order model at each level of the super categorical factor, each level should contain a center point, and we are able to estimate a full second-order model when collapsed over z.

Thus, we can see this is really an issue of having more than two levels. We certainly do have designs in our current catalogue that have whole plots of size 8 which is double the number of categorical factor levels in our super factor. So this likely could be accommodated more easily than three factor levels but we need to cognizant that these designs often take a long time to run and are not always computationally feasible to analyse.

It is clear that further research is appropriate to decide if adding a second categorical factor is a feasible option when runs are expensive as there seem to be many prohibitive restrictions. It may well be that this is not a good option unless a large number of runs is feasible and there are significant improvements to the computational time of the algorithm.

4.2.2 Unbalanced Designs

In chapter 2, we discussed a design in the literature called the minimum whole-plot design which was named as it was deemed the design with the least number of whole-plots for a central composite design. These designs, however, had a unique feature to them. The designs are both unbalanced (there are an odd number of runs per whole plot) and there is a row containing a center point within each whole plot. These features call for a unique approach to the assignment of the runs.

It is not difficult to see how one would, in theory, go about adding a categorical factor to this design. Instead of taking all vectors of size $nrpw$ with half equal to -1 and half equal to +1, we would take all vectors of size $nrpw$ where $\lfloor nrpw/2 \rfloor$ entries equal 1 and $\lceil nrpw/2 \rceil$ equal -1 and vice versa.

We would also need to add in additional checks to make sure there were not substantially more rows assigned to $z = +1$ or $z = -1$ and we would need to manually check that each level of z received one of the center points. Currently, this happens automatically as the whole plot with all the center points gets partitioned or doubled. This would not happen in the minimum whole plot design as the center points are dispersed throughout the design.

The main issue with assigning levels of the categorical factor within whole plots, however, is that the computational time would be greatly increased. For whole plots of size 5, there are 20 possible assignments of the categorical factor within a whole plot ($\binom{5}{3} \times 2$). For a design with 8 whole plots, there are $2.56 \times 20^{1}0$ assignments to test as compared to 1679616 ($=\binom{4}{2}^{8}$) for a design with only four runs per whole plot. As we were already running into issues with being able to compute the assignments globally for balanced designs, this issue could easily be exacerbated for unbalanced designs.

This implies that an important avenue of future research is finding ways to make the algorithm more computationally efficient. This could be done is cycling through the possible assignments in a way that we do not test every assignment plus its multiplicative inverse which is unnecessary as the multiplicative inverse will have the same value of the objective function for the D-criterion and will match the results of eligibility. It is not readily apparent how to do this in the way we cycle through the code. Solving the issue will, however, lead to a method that should be applicable in the balanced design setting as well which is highly desirable and allow us to potentially extend the algorithm to even bigger designs.

We can also see issues in unbalanced designs that, to use the splitting the whole-plot method, we will create a design where the whole-plots are of different sizes. It is possible this is not a desirable quality and it certainly makes analysis slightly more complicated. The way the D-optimality criterion's objective function is calculated would need to be revised in order to decide the size of each of the whole plots when creating the information matrix.

These issues are worthy of further research, not only on their own merit, but they should ultimately lead to making the original algorithm more computationally efficient and accommodating to a wider class of designs.

4.3 Conclusions

In this thesis, we bridged together two areas of response surface methodology: RSM designs with categorical factors and RSM split-plot designs. Though categorical factors are common in industrial experiments, the common designs for split-plot RSM are not suitable as is, for including categorical factors.

In 1988, John and Draper introduced a basis for RSM with categorical factors whereby they added a categorical factor to a pre-existing design that had desirable qualities for fitting a second-order model in quantitative factors.

This technique could not be applied directly to split-plot RSM designs because of the presence of randomization restrictions. In the thesis, we studied the structure of existing split-plot RSM designs in quantitative factors and developed strategies for adding on categorical factors while taking into account the randomization restrictions.

We thoroughly surveyed the conditions required for adding on the categorical factor to ensure that we were getting a feasible assignment, that is one that had appropriate design points to estimate a first-order design at each level of the categorical factor and had enough design points to be able to estimate a second-order design when collapsing over the categorical factor.

We found that often the feasible assignments were composed solely of assignments achieved through generating factors. The thesis established several results for the appropriate way to generate factors in a design with categorical factors. These rules apply to general screening designs as well as the response surface designs.

The methods we established in the thesis culminated in an algorithm that was implemented in R to find optimal assignments of a categorical factor to a given base design. We used the algorithm to produce tables of optimal assignments for quantitative split-plot RSM designs that are currently available in the literature.

From here, in this last chapter, we identified several avenues of future study for incorporating unbalanced design or multiple categorical factors. Work on these problems will also, by consequence, help improve the efficiency of the current algorithm. While we now have the appropriate designs for fitting these second-order models, it is also a next step to develop the appropriate tests for statistical inference on the presence of equal optimizing values at all levels of the categorical factors.

While there is future work to do, this thesis establishes the foundational results in design aspects for an important type of design that has not received previous treatment in the literature. The algorithm and tables will be of use to practitioners using categorical random variables in their split-plot designs but were previously restricted to designs that did not treat these factors with the care that was needed.

Appendix A

Tables of Example Designs for Algorithm

TABLE A.1. PKV 2007 OLG-GLS RS2020 design from Table A.1

X_1	X_2	x_1	x_2	X_1	X_2	x_1	x_2
-1	-1	-1	-1	$\sqrt{2}$	0	0	0
		1	-1			0	0
		-1	1			0	0
		1	1			0	0
1	-1	-1	-1	0	-$\sqrt{2}$	0	0
		1	1			0	0
		-1	-1			0	0
		1	1			0	0
-1	1	-1	-1	0	$\sqrt{2}$	0	0
		1	1			0	0
		-1	-1			0	0
		1	1			0	0
1	1	-1	-1	0	0	$-\sqrt{2}$	0
		1	1			$\sqrt{2}$	0
		-1	-1			0	$-\sqrt{2}$
		1	1			0	$\sqrt{2}$
-$\sqrt{2}$	0	0	0	0	0	0	0
		0	0			0	0
		0	0			0	0
		0	0			0	0

TABLE A.2. WKV 2009 OLS-GLS $RS3030$ design from Table 6

X_1	X_2	X_3	x_1	x_2	x_3	X_1	X_2	X_3	x_1	x_2	x_3
-1	-1	-1	-1	-1	-1	2	0	0	0	0	0
			1	1	-1				0	0	0
			1	-1	1				0	0	0
			-1	1	1				0	0	0
1	-1	-1	1	-1	-1	0	2	0	0	0	0
			-1	1	-1				0	0	0
			-1	-1	1				0	0	0
			1	1	1				0	0	0
-1	1	-1	1	-1	-1	0	-2	0	0	0	0
			-1	1	-1				0	0	0
			-1	-1	1				0	0	0
			1	1	1				0	0	0
1	1	-1	-1	-1	-1	0	0	2	0	0	0
			1	1	-1				0	0	0
			1	-1	1				0	0	0
			-1	1	1				0	0	0
-1	-1	1	1	-1	-1	0	0	-2	0	0	0
			-1	1	-1				0	0	0
			-1	-1	1				0	0	0
			1	1	1				0	0	0
1	-1	1	-1	-1	-1	0	0	0	$-\sqrt{2}$	0	0
			1	1	-1				$\sqrt{2}$	0	0
			1	-1	1				$-\sqrt{2}$	0	0
			-1	1	1				$\sqrt{2}$	0	0
-1	1	1	-1	-1	-1	0	0	0	0	$-\sqrt{2}$	0
			1	1	-1				0	$\sqrt{2}$	0
			1	-1	1				0	$-\sqrt{2}$	0
			-1	1	1				0	$\sqrt{2}$	0
1	1	1	1	-1	-1	0	0	0	0	0	$-\sqrt{2}$
			-1	1	-1				0	0	$\sqrt{2}$
			-1	-1	1				0	0	$-\sqrt{2}$
			1	1	1				0	0	$\sqrt{2}$
-2	0	0	0	0	0	0	0	0	0	0	0
			0	0	0				0	0	0
			0	0	0				0	0	0
			0	0	0				0	0	0

TABLE A.3. Factorial Points Design 24 WVK_2SR_D33

X_1	X_2	X_3	x_1	x_2	x_3	X_1	X_2	X_3	x_1	x_2	x_3
-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1
			-1	-1	1				-1	-1	1
			-1	1	-1				-1	1	-1
			-1	1	1				-1	1	1
			1	-1	-1				1	-1	-1
			1	-1	1				1	-1	1
			1	1	-1				1	1	-1
			1	1	1				1	1	1
-1	-1	1	-1	-1	-1	1	-1	1	-1	-1	-1
			-1	-1	1				-1	-1	1
			-1	1	-1				-1	1	-1
			-1	1	1				-1	1	1
			1	-1	-1				1	-1	-1
			1	-1	1				1	-1	1
			1	1	-1				1	1	-1
			1	1	1				1	1	1
-1	1	-1	-1	-1	-1	1	1	-1	-1	-1	-1
			-1	-1	1				-1	-1	1
			-1	1	-1				-1	1	-1
			-1	1	1				-1	1	1
			1	-1	-1				1	-1	-1
			1	-1	1				1	-1	1
			1	1	-1				1	1	-1
			1	1	1				1	1	1
-1	1	1	-1	-1	-1	1	1	1	-1	-1	-1
			-1	-1	1				-1	-1	1
			-1	1	-1				-1	1	-1
			-1	1	1				-1	1	1
			1	-1	-1				1	-1	-1
			1	-1	1				1	-1	1
			1	1	-1				1	1	-1
			1	1	1				1	1	1

Appendix B

Tables of Base Designs

B.1 Unbalanced VKM Method CCD

These designs are constructed us the Unbalanced Vining Kowalski Montgomery Method for CCD's, with the center point whole plot augmented with enough runs to become balanced and are OLS-GLS equivalent. The original designs can be found in the Parker catalog.

TABLE B.1. Design 1: VKMU_CCD_D12

X_1	x_1	x_2
-1	-1	-1
	1	-1
	-1	1
	1	1
1	-1	-1
	1	-1
	-1	1
	1	1
-1.73205	0	0
	0	0
	0	0
	0	0
1.73205	0	0
	0	0
	0	0
	0	0
0	-1.73205	0
	1.73205	0
	0	-1.73205
	0	1.73205
0	0	0
	0	0
	0	0
	0	0

TABLE B.2. Design 2: VKMU_CCD_D14

X_1	x_1	x_2	x_3	x_4	X_1	x_1	x_2	x_3	x_4
-1	-1	-1	-1	1	-2.23607	0	0	0	0
	1	-1	-1	-1		0	0	0	0
	-1	1	-1	-1		0	0	0	0
	1	1	-1	1		0	0	0	0
	-1	-1	1	-1		0	0	0	0
	1	-1	1	1		0	0	0	0
	-1	1	1	1		0	0	0	0
	1	1	1	-1		0	0	0	0
1	-1	-1	-1	-1	0	2.23607	0	0	0
	1	-1	-1	1		-2.23607	0	0	0
	-1	1	-1	1		0	2.23607	0	0
	1	1	-1	-1		0	-2.23607	0	0
	-1	-1	1	1		0	0	2.23607	0
	1	-1	1	-1		0	0	-2.23607	0
	-1	1	1	-1		0	0	0	2.23607
	1	1	1	1		0	0	0	-2.23607
2.23607	0	0	0	0	0	0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0

TABLE B.3. Design 3: VKMU_CCD_D22

X_1	X_2	x_1	x_2	X_1	X_2	x_1	x_2
-1	-1	-1	-1	2	0	0	0
		1	-1			0	0
		-1	1			0	0
		1	1			0	0
1	-1	-1	-1	0	-2	0	0
		1	-1			0	0
		-1	1			0	0
		1	1			0	0
-1	1	-1	-1	0	2	0	0
		1	-1			0	0
		-1	1			0	0
		1	1			0	0
1	1	-1	-1	0	0	-2	0
		1	-1			2	0
		-1	1			0	-2
		1	1			0	2
-2	0	0	0	0	0	0	0
		0	0			0	0
		0	0			0	0
		0	0			0	0

TABLE B.4. Design 4: VKMU_CCD_D24

X_1	X_2	x_1	x_2	x_3	x_4	X_1	X_2	x_1	x_2	x_3	x_4
-1	-1	-1	-1	-1	-1	-2.44949	0	0	0	0	0
		1	1	-1	-1			0	0	0	0
		1	-1	1	-1			0	0	0	0
		-1	1	1	-1			0	0	0	0
		1	-1	-1	1			0	0	0	0
		-1	1	-1	1			0	0	0	0
		-1	-1	1	1			0	0	0	0
		1	1	1	1			0	0	0	0
1		1	-1	-1	-1	0	2.44949	0	0	0	0
		-1	1	-1	-1			0	0	0	0
		-1	-1	1	-1			0	0	0	0
		1	1	1	-1			0	0	0	0
		-1	-1	-1	1			0	0	0	0
		1	1	-1	1			0	0	0	0
		1	-1	1	1			0	0	0	0
		-1	1	1	1			0	0	0	0
-1	1	1	-1	-1	-1	0	-2.44949	0	0	0	0
		-1	1	-1	-1			0	0	0	0
		-1	-1	1	-1			0	0	0	0
		1	1	1	-1			0	0	0	0
		-1	-1	-1	1			0	0	0	0
		1	1	-1	1			0	0	0	0
		1	-1	1	1			0	0	0	0
		-1	1	1	1			0	0	0	0
1	1	-1	-1	-1	-1	0	0	2.44949	0	0	0
		1	1	-1	-1			-2.44949	0	0	0
		1	-1	1	-1		0	0	2.44949	0	0
		-1	1	1	-1			0	-2.44949	0	0
		1	-1	-1	1			0	0	2.44949	0
		-1	1	-1	1			0	0	-2.44949	0
		-1	-1	1	1			0	0	0	2.44949
		1	1	1	1			0	0	0	-2.44949
2.44949	0	0	0	0	0	0	0	0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0

TABLE B.5. Design 5: VKMU_CCD_D32

X_1	X_2	X_3	x_1	x_2	X_1	X_2	X_3	x_1	x_2
-1	-1	-1	-1	1	2.23607	0	0	0	0
			1	-1				0	0
			-1	1				0	0
			1	-1				0	0
1	-1	-1	-1	-1	-2.23607	0	0	0	0
			1	1				0	0
			-1	-1				0	0
			1	1				0	0
-1	1	-1	-1	-1	0	2.23607	0	0	0
			1	1				0	0
			-1	-1				0	0
			1	1			0	0	0
1	1	-1	-1	1	0	-2.23607	0	0	0
			1	-1				0	0
			-1	1				0	0
			1	-1			0	0	0
-1	-1	1	-1	-1	0	0	2.23607	0	0
			1	1				0	0
			-1	-1				0	0
			1	1				0	0
1	-1	1	-1	1	0	0	-2.23607	0	0
			1	-1				0	0
			-1	1				0	0
			1	-1				0	0
-1	1	1	-1	1	0	0	0	2.23607	0
			1	-1			0	-2.23607	0
			-1	1				0	2.23607
			1	-1				0	-2.23607
1	1	1	-1	-1	0	0	0	0	0
			1	1				0	0
			-1	-1				0	0
			1	1				0	0

TABLE B.6. Design 6: VKMU_CCD_D34

X_1	X_2	X_3	x_1	x_2	x_3	x_4	X_1	X_2	X_3	x_1	x_2	x_3	x_4	X_1	X_2	X_3	x_1	x_2	x_3	x_4
-1	-1 -	1	-1	-1	-1	1	-1	1	1	-1	-1	-1	1	0	0	-2.64575	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
			1	1	-1	1				1	1	-1	1			-	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
			-1	1	1	1				-1	1	1	1				0	0	0	0
			1	1	1	-1				1	1	1	-1				0	0	0	0
1	-1	-1	-1	-1	-1	-	1	1	1	-1	-1	-1	-1	0	0	2.64575	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
			1	1	-1	-1				1	1	-1	-1				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
			-1	1	1	-1				-1	1	1	-1				0	0	0	0
			1	1	1	1				1	1	1	1				0	0	0	0
-1	1	-1	-1	-1	-1	-1	-2.64575	0	0	0	0	0	0	0	0	0	-2.64575	0	0	0
			1	-1	-1	1				0	0	0	0				2.64575	0	0	0
			-1	1	-1	1				0	0	0	0				0	-2.64575	0	0
			1	1	-1	-1				0	0	0	0				0	2.64575	0	0
			-1	-1	1	1				0	0	0	0				0	0	-2.64575	0
			1	-1	1	-1				0	0	0	0				0	0	2.64575	0
			-1	1	1	-1				0	0	0	0				0	0	0	-2.64575
			1	1	1	1				0	0	0	0				0	0	0	2.64575
1	1	-1	-1	-1	-1	1	2.64575	0	0	0	0	0	0	0	0	0	0	0	0	0
			1	-1	-1	-1				0	0	0	0				0	0	0	0
			-1	1	-1	-1				0	0	0	0				0	0	0	0
			1	1	-1	1				0	0	0	0				0	0	0	0
			-1	-1	1	-1				0	0	0	0				0	0	0	0
			1	-1	1	1				0	0	0	0				0	0	0	0
			-1	1	1	1				0	0	0	0				0	0	0	0
			1	1	1	-1				0	0	0	0				0	0	0	0
-1	-1	1	-1	-1	-1	-1	0	-2.64575	0	0	0	0	0							
			1	-1	-1	1				0	0	0	0							
			-1	1	-1	1				0	0	0	0							
			1	1	-1	-1				0	0	0	0							
			-1	-1	1	1				0	0	0	0							
			1	-1	1	-1				0	0	0	0							
			-1	1	1	-1				0	0	0	0							
			1	1	1	1				0	0	0	0							
1	-1	1	-1	-1	-1	1	0	2.64575	0	0	0	0	0							
			1	-1	-1	-1				0	0	0	0							
			-1	1	-1	-1				0	0	0	0							
			1	1	-1	1				0	0	0	0							
			-1	-1	1	-1				0	0	0	0							
			1	-1	1	1				0	0	0	0							
			-1	1	1	1				0	0	0	0							
			1	1	1	-1				0	0	0	0							

B.2 Balanced VKM Method CCD

These designs are constructed using the Balanced Vining Kowalski Montgomery Method for CCD's and are OLS-GLS equivalent. The original designs can be found in the Parker catalog.

TABLE B.7. Design Number 7: VKM_CCD_D12

X_1	x_1	x_2
-1	-1	-1
	1	-1
	-1	1
	1	1
1	-1	-1
	1	-1
	-1	1
	1	1
-1	0	0
	0	0
	0	0
	0	0
1	0	0
	0	0
	0	0
	0	0
0	-1	0
	1	0
	0	-1
	0	1
0	0	0
	0	0
	0	0
	0	0

TABLE B.8. Design 8: VKM_CCD_D13

X_1	x_1	x_2	x_3	X_1	x_1	x_2	x_3
-1	1	-1	-1	0	-1	0	0
	-1	1	-1		1	0	0
	-1	-1	1		-1	0	0
	1	1	1		1	0	0
	-1	-1	-1		-1	0	0
	1	1	-1		1	0	0
	1	-1	1		-1	0	0
	-1	1	1		1	0	0
1	1	-1	-1	0	0	-1	0
	-1	1	-1		0	1	0
	-1	-1	1		0	-1	0
	1	1	1		0	1	0
	-1	-1	-1		0	-1	0
	1	1	-1		0	1	0
	1	-1	1		0	-1	0
	-1	1	1		0	1	0
-1	0	0	0	0	0	0	-1
	0	0	0		0	0	1
	0	0	0		0	0	-1
	0	0	0		0	0	1
	0	0	0		0	0	-1
	0	0	0		0	0	1
	0	0	0		0	0	-1
	0	0	0		0	0	1
1	0	0	0	0	0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0

TABLE B.9. Design 9 VKM_CCD_D22

X_1	X_2	x_1	x_2	X_1	X_2	x_1	x_2
-1	-1	-1	-1	1	0	0	0
		1	-1			0	0
		-1	1			0	0
		1	1			0	0
1	-1	-1	-1	0	-1	0	0
		1	-1			0	0
		-1	1			0	0
		1	1			0	0
-1	1	-1	-1	0	1	0	0
		1	-1			0	0
		-1	1			0	0
		1	1			0	0
1	1	-1	-1	0	0	-1	0
		1	-1			1	0
		-1	1			0	-1
		1	1			0	1
-1	0	0	0	0	0	0	0
		0	0			0	0
		0	0			0	0
		0	0			0	0

TABLE B.10. Design 10 VKM_CCD_D23

X_1	X_2	x_1	x_2	x_3	X_1	X_2	x_1	x_2	x_3
-1	-1	-1	-1	1	0	-1	0	0	0
		1	-1	-1			0	0	0
		-1	1	-1			0	0	0
		1	1	1			0	0	0
1	-1	-1	-1	-1	0	1	0	0	0
		1	-1	1			0	0	0
		-1	1	1			0	0	0
		1	1	-1			0	0	0
-1	1	-1	-1	-1	0	0	-1	0	0
		1	-1	1			1	0	0
		-1	1	1			-1	0	0
		1	1	-1			1	0	0
1	1	-1	-1	1	0	0	0	-1	0
		1	-1	-1			0	1	0
		-1	1	-1			0	-1	0
		1	1	1			0	1	0
-1	0	0	0	0	0	0	0	0	-1
		0	0	0			0	0	1
		0	0	0			0	0	-1
		0	0	0			0	0	1
1	0	0	0	0	0	0	0	0	0
		0	0	0			0	0	0
		0	0	0			0	0	0
		0	0	0			0	0	0

TABLE B.11. Design 11: VKM_CCD_D24

X_1	X_2	x_1	x_2	x_3	x_4	X_1	X_2	x_1	x_2	x_3	x_4
-1	-1	-1	-1	-1	-1	1	0	0	0	0	0
		1	1	-1	-1			0	0	0	0
		1	-1	1	-1			0	0	0	0
		-1	1	1	-1			0	0	0	0
		1	-1	-1	1			0	0	0	0
		-1	1	-1	1			0	0	0	0
		-1	-1	1	1			0	0	0	0
		1	1	1	1			0	0	0	0
1	-1	1	-1	-1	-1	0	-1	0	0	0	0
		-1	1	-1	-1			0	0	0	0
		-1	-1	1	-1			0	0	0	0
		1	1	1	-1			0	0	0	0
		-1	-1	-1	1			0	0	0	0
		1	1	-1	1			0	0	0	0
		1	-1	1	1			0	0	0	0
		-1	1	1	1			0	0	0	0
-1	1	1	-1	-1	-1	0	1	0	0	0	0
		-1	1	-1	-1			0	0	0	0
		-1	-1	1	-1			0	0	0	0
		1	1	1	-1			0	0	0	0
		-1	-1	-1	1			0	0	0	0
		1	1	-1	1			0	0	0	0
		1	-1	1	1			0	0	0	0
		-1	1	1	1			0	0	0	0
1	1	-1	-1	-1	-1	0	0	-1	0	0	0
		1	1	-1	-1			1	0	0	0
		1	-1	1	-1			0	-1	0	0
		-1	1	1	-1			0	1	0	0
		1	-1	-1	1			0	0	-1	0
		-1	1	-1	1			0	0	1	0
		-1	-1	1	1			0	0	0	-1
		1	1	1	1			0	0	0	1
-1	0	0	0	0	0	0	0	0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0

TABLE B.12. Design 12: VKM_CCD_D32

X_1	X_2	X_3	x_1	x_2	X_1	X_2	X_3	x_1	x_2
-1	-1	-1	-1	1	-1	0	0	0	0
			1	-1				0	0
			-1	1				0	0
			1	-1				0	0
1	-1	-1	-1	-1	1	0	0	0	0
			1	1				0	0
			-1	-1				0	0
			1	1				0	0
-1	1	-1	-1	-1	0	-1	0	0	0
			1	1				0	0
			-1	-1				0	0
			1	1				0	0
1	1	-1	-1	1	0	1	0	0	0
			1	-1				0	0
			-1	1				0	0
			1	-1				0	0
-1	-1	1	-1	-1	0	0	-1	0	0
			1	1				0	0
			-1	-1				0	0
			1	1				0	0
1	-1	1	-1	1	0	0	1	0	0
			1	-1				0	0
			-1	1				0	0
			1	-1				0	0
-1	1	1	-1	1	0	0	0	-1	0
			1	-1				1	0
			-1	1				0	-1
			1	-1				0	1
1	1	1	-1	-1	0	0	0	0	0
			1	1				0	0
			-1	-1				0	0
			1	1				0	0

TABLE B.13. Design 13 : VKM_CCD_D33

X_1	X_2	X_3	x_1	x_2	x_3	X_1	X_2	X_3	x_1	x_2	x_3
-1	-1	-1	-1	-1	-1	1	0	0	0	0	0
			1	1	-1				0	0	0
			1	-1	1				0	0	0
			-1	1	1				0	0	0
1	-1	-1	1	-1	-1	0	1	0	0	0	0
			-1	1	-1				0	0	0
			-1	-1	1				0	0	0
			1	1	1				0	0	0
-1	1	-1	1	-1	-1	0	-1	0	0	0	0
			-1	1	-1				0	0	0
			-1	-1	1				0	0	0
			1	1	1				0	0	0
1	1	-1	-1	-1	-1	0	0	1	0	0	0
			1	1	-1				0	0	0
			1	-1	1				0	0	0
			-1	1	1				0	0	0
-1	-1	1	1	-1	-1	0	0	-1	0	0	0
			-1	1	-1				0	0	0
			-1	-1	1				0	0	0
			1	1	1				0	0	0
1	-1	1	-1	-1	-1	0	0	0	-1	0	0
			1	1	-1				1	0	0
			1	-1	1				-1	0	0
			-1	1	1				1	0	0
-1	1	1	-1	-1	-1	0	0	0	0	-1	0
			1	1	-1				0	1	0
			1	-1	1				0	-1	0
			-1	1	1				0	1	0
1	1	1	1	-1	-1	0	0	0	0	0	-1
			-1	1	-1				0	0	1
			-1	-1	1				0	0	-1
			1	1	1				0	0	1
-1	0	0	0	0	0	0	0	0	0	0	0
			0	0	0				0	0	0
			0	0	0				0	0	0
			0	0	0				0	0	0

TABLE B.14. Design 14: VKM_CCD_D34

X_1	X_2	X_3	x_1	x_2	x_3	x_4	X_1	X_2	X_3	x_1	x_2	x_3	x_4	X_1	X_2	X_3	x_1	x_2	x_3	x_4
-1	-1	-1	-1	-1	-1	1	-1	1	1	-1	-1	-1	1	0	0	-1	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
			1	1	-1	1				1	1	-1	1				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
			-1	1	1	1				-1	1	1	1				0	0	0	0
			1	1	1	-1				1	1	1	-1				0	0	0	0
1	-1	-1	-1	-1	-1	-1	1	1	1	-1	-1	-1 -1	0	0	1	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
			1	1	-1	-1				1	1	-1	-1				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
			-1	1	1	-1				-1	1	1	-1				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	0	0	0	0	0	-1	0	0	0
			1	-1	-1	1				0	0	0	0				1	0	0	0
			-1	1	-1	1				0	0	0	0				0	-1	0	0
			1	1	-1	-1				0	0	0	0				0	1	0	0
			-1	-1	1	1				0	0	0	0				0	0	-1	0
			1	-1	1	-1				0	0	0	0				0	0	1	0
			-1	1	1	-1				0	0	0	0				0	0	0	-1
			1	1	1	1				0	0	0	0				0	0	0	1
1	1	-1	-1	-1	-1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
			1	-1	-1	-1				0	0	0	0				0	0	0	0
			-1	1	-1	-1				0	0	0	0				0	0	0	0
			1	1	-1	1				0	0	0	0				0	0	0	0
			-1	-1	1	-1				0	0	0	0				0	0	0	0
			1	-1	1	1				0	0	0	0				0	0	0	0
			-1	1	1	1				0	0	0	0				0	0	0	0
			1	1	1	-1				0	0	0	0				0	0	0	0
-1	-1	1	-1	-1	-1	-1	0	-1	0	0	0	0	0							
			1	-1	-1	1				0	0	0	0							
			-1	1	-1	1				0	0	0	0							
			1	1	-1	-1				0	0	0	0							
			-1	-1	1	1				0	0	0	0							
			1	-1	1	-1				0	0	0	0							
			-1	1	1	-1				0	0	0	0							
			1	1	1	1				0	0	0	0							
1	-1	1	-1	-1	-1	1	0	1	0	0	0	0	0							
			1	-1	-1	-1				0	0	0	0							
			-1	1	-1	-1				0	0	0	0							
			1	1	-1	1				0	0	0	0							
			-1	-1	1	-1				0	0	0	0							
			1	-1	1	1				0	0	0	0							
			-1	1	1	1				0	0	0	0							
			1	1	1	-1				0	0	0	0							

B.3 Two-Strata Rotatable Designs

These designs are two-strata rotatable and are taken from the paper Wang, Vining, and Kowalski (2010).

TABLE B.15. Design 15: WVK_2SR_D12

X_1	x_1	x_2
-1	-1	-1
-1	-1	1
-1	1	-1
-1	1	1
1	-1	-1
1	-1	1
1	1	-1
1	1	1
-1.89	0	0
-1.89	0	0
-1.89	0	0
-1.89	0	0
1.89	0	0
1.89	0	0
1.89	0	0
1.89	0	0
0	-1.682	0
0	1.682	0
0	0	-1.682
0	0	1.682
0	0	0
0	0	0
0	0	0
0	0	0

TABLE B.16. Design 16: WVK_2SR_D13

X_1	x_1	x_2	x_3	X_1	x_1	x_2	x_3
-1	-1	-1	-1	0	-1.565085	0	0
	-1	-1	1		1.565085	0	0
	-1	1	-1		-1.565085	0	0
	-1	1	1		1.565085	0	0
	1	-1	-1		-1.565085	0	0
	1	-1	1		1.565085	0	0
	1	1	-1		-1.565085	0	0
	1	1	1		1.565085	0	0
1	-1	-1	-1	0	0	-1.565085	0
	-1	-1	1		0	1.565085	0
	-1	1	-1		0	-1.565085	0
	-1	1	1		0	1.565085	0
	1	-1	-1		0	-1.565085	0
	1	-1	1		0	1.565085	0
	1	1	-1		0	-1.565085	0
	1	1	1		0	1.565085	0
-1.189	0	0	0	0	0	0	-1.565085
	0	0	0		0	0	1.565085
	0	0	0		0	0	-1.565085
	0	0	0		0	0	1.565085
	0	0	0		0	0	-1.565085
	0	0	0		0	0	1.565085
	0	0	0		0	0	-1.565085
	0	0	0		0	0	1.565085
1.189	0	0	0	0	0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0

TABLE B.17. Design 17: WVK_2SR_D14

X_1	x_1	x_2	x_3	x_4	X_1	x_1	x_2	x_3	x_4
-1	-1	-1	-1	1	1.189	0	0	0	0
	-1	-1	1	-1		0	0	0	0
	-1	1	-1	-1		0	0	0	0
	-1	1	1	1		0	0	0	0
	1	-1	-1	-1		0	0	0	0
	1	-1	1	1		0	0	0	0
	1	1	-1	1		0	0	0	0
	1	1	1	-1		0	0	0	0
	-1	-1	-1	-1	0	-2	0	0	0
	-1	-1	1	1		2	0	0	0
	-1	1	-1	1		0	-2	0	0
	-1	1	1	-1		0	2	0	0
	1	-1	-1	1		0	0	-2	0
	1	-1	1	-1		0	0	2	0
	1	1	-1	-1		0	0	0	-2
	1	1	1	1		0	0	0	2
-1.189	0	0	0	0	0	0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0
	0	0	0	0		0	0	0	0

TABLE B.18. Design 18: WVK_2SR_D21

X_1	X_2	x_1
-1	-1	-1
		1
-1	1	-1
		1
1	-1	-1
		1
1	1	-1
		1
-1.189	0	0
		0
1.189	0	0
		0
0	-1.189	0
		0
0	1.189	0
		0
0	0	-6.928203
		6.928203
0	0	0
		0
0	0	0
		0
0	0	0
		0

TABLE B.19. Design 19: WVK_2SR_D22

X_1	X_2	x_1	x_2	X_1	X_2	x_1	x_2
-1	-1	-1	-1	1.414	0	0	0
		-1	1			0	0
		1	-1			0	0
		1	1			0	0
-1	1	-1	-1	0	-1.414	0	0
		-1	1			0	0
		1	-1			0	0
		1	1			0	0
1	-1	-1	-1	0	1.414	0	0
		-1	1			0	0
		1	-1			0	0
		1	1			0	0
1	1	-1	-1	0	0	-2	0
		-1	1			2	0
		1	-1			0	-2
		1	1			0	2
-1.414	0	0	0	0	0	0	0
		0	0			0	0
		0	0			0	0
		0	0			0	0

TABLE B.20. Design 20: WVK_2SR_D23

X_1	X_2	x_1	x_2	x_3	X_1	X_2	x_1	x_2	x_3
-1	-1	-1	-1	-1	0	-1.414	0	0	0
		-1	-1	1			0	0	0
		-1	1	-1			0	0	0
		-1	1	1			0	0	0
		1	-1	-1			0	0	0
		1	-1	1			0	0	0
		1	1	-1			0	0	0
		1	1	1			0	0	0
-1	1	-1	-1	-1	0	1.414	0	0	0
		-1	-1	1			0	0	0
		-1	1	-1			0	0	0
		-1	1	1			0	0	0
		1	-1	-1			0	0	0
		1	-1	1			0	0	0
		1	1	-1			0	0	0
		1	1	1			0	0	0
1	-1	-1	-1	-1	0	0	-2.514867	0	0
		-1	-1	1			2.514867	0	0
		-1	1	-1			-2.514867	0	0
		-1	1	1			2.514867	0	0
		1	-1	-1			-2.514867	0	0
		1	-1	1			2.514867	0	0
		1	1	-1			-2.514867	0	0
		1	1	1			2.514867	0	0
1	1	-1	-1	-1	0	0	0	-2.514867	0
		-1	-1	1			0	2.514867	0
		-1	1	-1			0	-2.514867	0
		-1	1	1			0	2.514867	0
		1	-1	-1			0	-2.514867	0
		1	-1	1			0	2.514867	0
		1	1	-1			0	-2.514867	0
		1	1	1			0	2.514867	0
-1.414	0	0	0	0	0	0	0	0	-2.514867
		0	0	0			0	0	2.514867
		0	0	0			0	0	-2.514867
		0	0	0			0	0	2.514867
		0	0	0			0	0	-2.514867
		0	0	0			0	0	2.514867
		0	0	0			0	0	-2.514867
		0	0	0			0	0	2.514867
1.414	0	0	0	0	0	0	0	0	0
		0	0	0			0	0	0
		0	0	0			0	0	0
		0	0	0			0	0	0
		0	0	0			0	0	0
		0	0	0			0	0	0
		0	0	0			0	0	0
		0	0	0			0	0	0

TABLE B.21. Design 21: WVK_2SR_D24

X_1	X_2	x_1	x_2	x_3	x_4	X_1	X_2	x_1	x_2	x_3	x_4
-1		-1	-1	-1	1	1.414	0	0	0	0	0
		-1	-1	1	-1			0	0	0	0
		-1	1	-1	-1			0	0	0	0
		-1	1	1	1			0	0	0	0
		1	-1	-1	-1			0	0	0	0
		1	-1	1	1			0	0	0	0
		1	1	-1	1			0	0	0	0
		1	1	1	-1			0	0	0	0
-1	1	-1	-1	-1	1	0	-1.414	0	0	0	0
		-1	-1	1	-1			0	0	0	0
		-1	1	-1	-1			0	0	0	0
		-1	1	1	1			0	0	0	0
		1	-1	-1	-1			0	0	0	0
		1	-1	1	1			0	0	0	0
		1	1	-1	1			0	0	0	0
		1	1	1	-1			0	0	0	0
1	-1	-1	-1	-1	-1	0	1.414	0	0	0	0
		-1	-1	1	1			0	0	0	0
		-1	1	-1	1			0	0	0	0
		-1	1	1	-1			0	0	0	0
		1	-1	-1	1			0	0	0	0
		1	-1	1	-1			0	0	0	0
		1	1	-1	-1			0	0	0	0
		1	1	1	1			0	0	0	0
1	1	-1	-1	-1	-1	0	0	-2.378	0	0	0
		-1	-1	1	1			2.378	0	0	0
		-1	1	-1	1			0	-2.378	0	0
		-1	1	1	-1			0	2.378	0	0
		1	-1	-1	1			0	0	-2.378	0
		1	-1	1	-1			0	0	2.378	0
		1	1	-1	-1			0	0	0	-2.378
		1	1	1	1			0	0	0	2.378
-1.414	0	0	0	0	0	0	0	0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0

TABLE B.22. Design 22: WVK_2SR_D31

X_1	X_2	X_3	
-1	-1	-1	-1
			1
-1	-1	1	-1
			1
-1	1	-1	-1
			1
-1	1	1	-1
			1
1	-1	-1	-1
			1
1	-1	1	-1
			1
1	1	-1	-1
			1
1	1	1	-1
			1
-1.682	0	0	0
			0
1.682	0	0	0
			0
0	-1.682	0	0
			0
0	1.682	0	0
			0
0	0	-1.682	0
			0
0	0	1.682	0
			0
0	0	0	-21.05718
			21.05718
0	0	0	0
			0

TABLE B.23. Design 23: WVK_2SR_D32

X_1	X_2	X_3	x_1	x_2	X_1	X_2	X_3	x_1	x_2
-1	-1	-1	-1	-1	-1.682	0	0	0	0
			-1	1				0	0
			1	-1				0	0
			1	1				0	0
-1	-1	1	-1	-1	1.682	0	0	0	0
			-1	1				0	0
			1	-1				0	0
			1	1				0	0
-1	1	-1	-1	-1	0	-1.682	0	0	0
			-1	1				0	0
			1	-1				0	0
			1	1				0	0
-1	1	1	-1	-1	0	1.682	0	0	0
			-1	1				0	0
			1	-1				0	0
			1	1				0	0
1	-1	-1	-1	-1	0	0	-1.682	0	0
			-1	1				0	0
			1	-1				0	0
			1	1				0	0
1	-1	1	-1	-1	0	0	1.682	0	0
			-1	1				0	0
			1	-1				0	0
			1	1				0	0
1	1	-1	-1	-1	0	0	0	-2.378	0
			-1	1				2.378	0
			1	-1				0	-2.378
			1	1				0	2.378
1	1	1	-1	-1	0	0	0	0	0
			-1	1				0	0
			1	-1				0	0
			1	1				0	0

TABLE B.24. Design 24: WVK_2SR_D33

X_1	X_2	X_3	x_1	x_2	x_3	X_1	X_2	X_3	x_1	x_2	x_3	X_1	X_2	X_3	x_1	x_2	x_3
-1	-1	-1	-1	-1	-1	1	1	-1	-1	-1	-1	0	0	-1.682	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				0	0	0
-1	-1	-1	1	-1	-1	1	1	-1	1	-1	-1	0	0	-1.682	0	0	0
	-1	-1	1	-1	1				1	-1	1				0	0	0
	-1	-1	1	1	-1				1	1	-1				0	0	0
	-1	-1	1	1	1				1	1	1				0	0	0
-1	-1	1	-1	-1	-1	1	1	1	-1	-1	-1	0	0	1.682	0	0	0
	-1	1	-1	-1	1				-1	-1	1				0	0	0
	-1	1	-1	1	-1				-1	1	-1				0	0	0
	-1	1	-1	1	1				-1	1	1				0	0	0
-1	-1	1	1	-1	-1	1	1	1	1	-1	-1	0	0	1.682	0	0	0
	-1	1	1	-1	1				1	-1	1				0	0	0
	-1	1	1	1	-1				1	1	-1				0	0	0
	-1	1	1	1	1				1	1	1				0	0	0
-1	1	-1	-1	-1	-1	-1.682	0	0	0	0	0	0	0	0	-2.990698	0	0
	1	-1	-1	-1	1				0	0	0				2.990698	0	0
	1	-1	-1	1	-1				0	0	0				-2.990698	0	0
	1	-1	-1	1	1				0	0	0				2.990698	0	0
-1	1	-1	1	-1	-1	-1.682	0	0	0	0	0	0	0	0	-2.990698	0	0
	1	-1	1	-1	1				0	0	0				2.990698	0	0
	1	-1	1	1	-1				0	0	0				-2.990698	0	0
	1	-1	1	1	1				0	0	0				2.990698	0	0
-1	1	1	-1	-1	-1	1.682	0	0	0	0	0	0	0	0	0	-2.990698	0
	1	1	-1	-1	1				0	0	0				0	2.990698	0
	1	1	-1	1	-1				0	0	0				0	-2.990698	0
	1	1	-1	1	1				0	0	0				0	2.990698	0
-1	1	1	1	-1	-1	1.682	0	0	0	0	0	0	0	0	0	-2.990698	0
	1	1	1	-1	1				0	0	0				0	2.990698	0
	1	1	1	1	-1				0	0	0				0	-2.990698	0
	1	1	1	1	1				0	0	0				0	2.990698	0
1	-1	-1	-1	-1	-1	0	-1.682	0	0	0	0	0	0	0	0	0	-2.990698
	-1	-1	-1	-1	1				0	0	0				0	0	2.990698
	-1	-1	-1	1	-1				0	0	0				0	0	-2.990698
	-1	-1	-1	1	1				0	0	0				0	0	2.990698
1	-1	-1	1	-1	-1	0	-1.682	0	0	0	0	0	0	0	0	0	-2.990698
	-1	-1	1	-1	1				0	0	0				0	0	2.990698
	-1	-1	1	1	-1				0	0	0				0	0	-2.990698
	-1	-1	1	1	1				0	0	0				0	0	2.990698
1	-1	1	-1	-1	-1	0	1.682	0	0	0	0	0	0	0	0	0	0
	-1	1	-1	-1	1				0	0	0				0	0	0
	-1	1	-1	1	-1				0	0	0				0	0	0
	-1	1	-1	1	1				0	0	0				0	0	0
1	-1	1	1	-1	-1	0	1.682	0	0	0	0	0	0	0	0	0	0
	-1	1	1	-1	1				0	0	0				0	0	0
	-1	1	1	1	-1				0	0	0				0	0	0
	-1	1	1	1	1				0	0	0				0	0	0

TABLE B.25. Design 25: WVK_2SR_D34

X_1	X_2	X_3	x_1	x_2	x_3	x_4	X_1	X_2	X_3	x_1	x_2	x_3	x_4	X_1	X_2	X_3	x_1	x_2	x_3	x_4
-1	-1	-1	-1	-1	-1	1	1	1	-1	-1	-1	-1	1	0	0	-1.682	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
			-1	1	1	1				-1	1	1	1				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
			1	1	-1	1				1	1	-1	1				0	0	0	0
			1	1	1	-1				1	1	1	-1				0	0	0	0
-1	-1	1	-1	-1	-1	-1	1	1	1	-1	-1	-1	-1	0	0	1.682	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
			-1	1	1	-1				-1	1	1	-1				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
			1	1	-1	-1				1	1	-1	-1				0	0	0	0
			1	1	1	1				1	1	1	1				0	0	0	0
-1			-1	-1	-1	-1	-1.682	0	0	0	0	0	0	0	0	0	-2.828	0	0	0
			-1	-1	1	1				0	0	0	0				2.828	0	0	0
			-1	1	-1	1				0	0	0	0				0	-2.828	0	0
			-1	1	1	-1				0	0	0	0				0	2.828	0	0
			1	-1	-1	1				0	0	0	0				0	0	-2.828	0
			1	-1	1	-1				0	0	0	0				0	0	2.828	0
			1	1	-1	-1				0	0	0	0				0	0	0	-2.828
			1	1	1	1				0	0	0	0				0	0	0	2.828
-1	1	1	-1	-1	-1	1	1.682	0	0	0	0	0	0	0	0	0	0	0	0	0
			-1	-1	1	-1				0	0	0	0				0	0	0	0
			-1	1	-1	-1				0	0	0	0				0	0	0	0
			-1	1	1	1				0	0	0	0				0	0	0	0
			1	-1	-1	-1				0	0	0	0				0	0	0	0
			1	-1	1	1				0	0	0	0				0	0	0	0
			1	1	-1	1				0	0	0	0				0	0	0	0
			1	1	1	-1				0	0	0	0				0	0	0	0
1	-1	-1	-1	-1	-1	-1	0	-1.682	0	0	0	0	0							
			-1	-1	1	1				0	0	0	0							
			-1	1	-1	1				0	0	0	0							
			-1	1	1	-1				0	0	0	0							
			1	-1	-1	1				0	0	0	0							
			1	-1	1	-1				0	0	0	0							
			1	1	-1	-1				0	0	0	0							
			1	1	1	1				0	0	0	0							
1	-1	1	-1	-1	-1	1	0	1.682	0	0	0	0	0							
			-1	-1	1	-1				0	0	0	0							
			-1	1	-1	-1				0	0	0	0							
			-1	1	1	1				0	0	0	0							
			1	-1	-1	-1				0	0	0	0							
			1	-1	1	1				0	0	0	0							
			1	1	-1	1				0	0	0	0							
			1	1	1	-1				0	0	0	0							

B.4 CUBE Designs

These designs are CUBE designs and are constructed following the directions in Draper and John (1998).

TABLE B.26. Design 26: CUBE_D12

X_1	x_1	x_2
-1	-0.82	-0.82
	-0.82	0.82
	0.82	-0.82
	0.82	0.82
1	-0.82	-0.82
	-0.82	0.82
	0.82	-0.82
	0.82	0.82
0	-1.5	0
	1.5	0
	0	-1.5
	0	1.5
0	0	0
	0	0
	0	0
	0	0

TABLE B.27. Design 27: CUBE_D22

X_1	X_2	x_1	x_2
-1	-1	-0.82	-0.82
		-0.82	0.82
		0.82	-0.82
		0.82	0.82
-1	1	-0.82	-0.82
		-0.82	0.82
		0.82	-0.82
		0.82	0.82
1	-1	-0.82	-0.82
		-0.82	0.82
		0.82	-0.82
		0.82	0.82
1	1	-0.82	-0.82
		-0.82	0.82
		0.82	-0.82
		0.82	0.82
0	0	-1.94	0.00
		1.94	0.00
		0.00	-1.94
		0.00	1.94
0	0	0.00	0.00
		0.00	0.00
		0.00	0.00
		0.00	0.00

TABLE B.28. Design 28: CUBE_D32

X_1	X_2	X_3	x_1	x_2	X_1	X_2	X_3	x_1	x_2
-1	-1	-1	-0.84	-0.84	1	-1	1	-0.84	-0.84
			-0.84	0.84				-0.84	0.84
			0.84	-0.84				0.84	-0.84
			0.84	0.84				0.84	0.84
-1	-1	1	-0.84	-0.84	1	1	-1	-0.84	-0.84
			-0.84	0.84				-0.84	0.84
			0.84	-0.84				0.84	-0.84
			0.84	0.84				0.84	0.84
-1	1	-1	-0.84	-0.84	1	1	1	-0.84	-0.84
			-0.84	0.84				-0.84	0.84
			0.84	-0.84				0.84	-0.84
			0.84	0.84				0.84	0.84
-1	1	1	-0.84	-0.84	0	0	0	-2.44	0.00
			-0.84	0.84				2.44	0.00
			0.84	-0.84				0.00	-2.44
			0.84	0.84				0.00	2.44
1	-1	-1	-0.84	-0.84	0	0	0	0.00	0.00
			-0.84	0.84				0.00	0.00
			0.84	-0.84				0.00	0.00
			0.84	0.84				0.00	0.00

TABLE B.29. Design 29: CUBE_D42

X_1	X_2	X_3	X_4	x_1	x_2	X_1	X_2	X_3	X_4	x_1	x_2
-1	-1	-1	-1	-0.87	-0.87	1	-1	-1	1	-0.87	-0.87
				-0.87	0.87					-0.87	0.87
				0.87	-0.87					0.87	-0.87
				0.87	0.87					0.87	0.87
-1	-1	-1	1	-0.87	-0.87	1	-1	1	-1	-0.87	-0.87
				-0.87	0.87					-0.87	0.87
				0.87	-0.87					0.87	-0.87
				0.87	0.87					0.87	0.87
-1	-1	1	-1	-0.87	-0.87	1	-1	1	1	-0.87	-0.87
				-0.87	0.87					-0.87	0.87
				0.87	-0.87					0.87	-0.87
				0.87	0.87					0.87	0.87
-1	-1	1	1	-0.87	-0.87	1	1	-1	-1	-0.87	-0.87
				-0.87	0.87					-0.87	0.87
				0.87	-0.87					0.87	-0.87
				0.87	0.87					0.87	0.87
-1	1	-1	-1	-0.87	-0.87	1	1	-1	1	-0.87	-0.87
				-0.87	0.87					-0.87	0.87
				0.87	-0.87					0.87	-0.87
				0.87	0.87					0.87	0.87
-1	1	-1	1	-0.87	-0.87	1	1	1	-1	-0.87	-0.87
				-0.87	0.87					-0.87	0.87
				0.87	-0.87					0.87	-0.87
				0.87	0.87					0.87	0.87
-1	1	1	-1	-0.87	-0.87	1	1	1	1	-0.87	-0.87
				-0.87	0.87					-0.87	0.87
				0.87	-0.87					0.87	-0.87
				0.87	0.87					0.87	0.87
-1	1	1	1	-0.87	-0.87	0	0	0	0	-3.03	0
				-0.87	0.87					3.03	0
				0.87	-0.87					0	-3.03
				0.87	0.87					0	3.03
1	-1	-1	-1	-0.87	-0.87	0	0	0	0	0	0
				-0.87	0.87					0	0
				0.87	-0.87					0	0
				0.87	0.87					0	0

B.5 Balanced VKM Method SCD

These designs are constructed by the Balanced Vining, Kowalski, and Montgomery Method for Small Composite Designs and are OLS-GLS equivalent. They can be found in the Parker Catalog.

TABLE B.30. Design 30: VKM_SCD_S12

X_1	x_1	x_2
-1	1	1
	1	1
	-1	-1
	-1	-1
1	1	-1
	1	-1
	-1	1
	-1	1
-1	0	0
	0	0
	0	0
	0	0
0	-1	0
	1	0
	0	-1
	0	1
1	0	0
	0	0
	0	0
	0	0
0	0	0
	0	0
	0	0
	0	0

TABLE B.31. Design 31: VKM_SCD_D13

X_1	x_1	x_2	x_3	X_1	x_1	x_2	x_3
-1	1	-1	1	1.68	-1.68	0	0
	1	-1	1		-1.68	0	0
	1	-1	1		1.68	0	0
	-1	1	-1		1.68	0	0
	-1	1	-1		0	-1.68	0
	-1	1	-1		0	-1.68	0
	1	1	1		0	1.68	0
	1	1	1		0	1.68	0
	1	1	1		0	0	-1.68
	-1	-1	-1		0	0	-1.68
	-1	-1	-1		0	0	1.68
	-1	-1	-1		0	0	1.68
1	1	1	-1	0	0	0	0
	1	1	-1		0	0	0
	1	1	-1		0	0	0
	1	-1	-1		0	0	0
	1	-1	-1		0	0	0
	1	-1	-1		0	0	0
	-1	1	1		0	0	0
	-1	1	1		0	0	0
	-1	1	1		0	0	0
	-1	-1	1		0	0	0
	-1	-1	1		0	0	0
	-1	-1	1		0	0	0
-1.68	0	0	0	0	0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0
	0	0	0		0	0	0

TABLE B.32. Design 32: VKM_SCD_D22

X_1	X_2	x_1	x_2	X_1	X_2	x_1	x_2
-1	-1	1	-1	0	1.68	0	0
		1	-1			0	0
		-1	-1			0	0
		-1	-1			0	0
-1	1	-1	1	0	0	1.68	0
		-1	1			-1.68	0
		1	1			0	1.68
		1	1			0	-1.68
1	-1	1	1	0	-1.68	0	0
		1	1			0	0
		-1	1			0	0
		-1	1			0	0
1	1	1	-1	-1.68	0	0	0
		1	-1			0	0
		-1	-1			0	0
		-1	-1			0	0
1.68	0	0	0	0	0	0	0
		0	0			0	0
		0	0			0	0
		0	0			0	0

TABLE B.33. Design 33: VKM_SCD_D24

X_1	X_2	x_1	x_2	x_3	x_4	X_1	X_2	x_1	x_2	x_3	x_4
-1	-1	1	-1	-1	1	1	0	0	0	0	0
		-1	1	1	-1			0	0	0	0
		1	1	1	1		0	0	0	0	0
		-1	-1	-1	-1			0	0	0	0
		1	-1	-1	1			0	0	0	0
		-1	1	1	-1			0	0	0	0
		1	1	1	1			0	0	0	0
		-1	-1	-1	-1			0	0	0	0
1	-1	1	-1	1	1	0	-1	0	0	0	0
		1	1	-1	1			0	0	0	0
		-1	1	-1	-1			0	0	0	0
		-1	-1	1	-1			0	0	0	0
		1	-1	1	1			0	0	0	0
		1	1	-1	1			0	0	0	0
		-1	1	-1	-1			0	0	0	0
		-1	-1	1	-1			0	0	0	0
-1	1	-1	1	1 1	0	1	0	0	0	0	
		1	-1	-1	-1			0	0	0	0
		-1	-1	-1	1			0	0	0	0
		1	1	1	-1			0	0	0	0
		-1	1	1	1			0	0	0	0
		1	-1	-1	-1			0	0	0	0
		-1	-1	-1	1			0	0	0	0
		1	1	1	-1			0	0	0	0
1	1	1	1	-1	-1	0	0	-1	0	0	0
		1	-1	1	-1			1	0	0	0
		-1	1	-1	1			0	-1	0	0
		-1	-1	1	1			0	1	0	0
		1	1	-1	-1			0	0	-1	0
		1	-1	1	-1			0	0	1	0
		-1	1	-1	1			0	0	0	-1
		-1	-1	1	1			0	0	0	1
-1	0	0	0	0	0	0	0	0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0
		0	0	0	0			0	0	0	0

TABLE B.34. Design 34: VKM_SCD_D34

X_1	X_2	X_3	x_1	x_2	X_1	X_2	X_3	x_1	x_2
-1	-1	-1	-1	-1	-1.82	0	0	0	0
		-	-1	-1				0	0
			-1	-1				0	0
			-1	-1				0	0
-1	-1	1	1	1	0	-1.82	0	0	0
			1	1				0	0
			1	1				0	0
			1	1				0	0
-1	1	-1	1	1	0	0	-1.82	0	0
			1	1				0	0
			1	1				0	0
			1	1				0	0
-1	1	1	-1	1	0	0	0	-1.82	0
			-1	1				1.82	0
			1	-1				0	-1.82
			1	-1				0	1.82
1	-1	-1	1	1	0	0	1.82	0	0
			1	1				0	0
			1	1				0	0
			1	1				0	0
1	-1	1	1	-1	0	1.82	0	0	0
			1	-1				0	0
			-1	1				0	0
			-1	1				0	0
1	1	-1	1	-1	1.82	0	0	0	0
			1	-1				0	0
			-1	1				0	0
			-1	1				0	0
1	1	1	-1	-1	0	0	0	0	0
			-1	-1				0	0
			-1	-1				0	0
			-1	-1				0	0

Appendix C

Algorithm Code

```
#Required Packages:
#gtools: permutations
#Deducer
############# Algorithm Inputs #####################
# XQ:  Design matrix in the factorial points
# nwp: Number of WPs
# nfwp: Number of WPs in the factorial points
# nrpw: Number of runs per WP
# nwx: Number of quantitative WP factors
# method: Type of partitioning:
#         1: Add SP
#         2: Add WP by pseudo partitioning
#         3: Add WP by splitting WPs
#         4: Add WP by doubling entire design

algorithm<-function(XQ, nwp, nfwp, nrpw, nwx, method){
```

```
  ######### PRE-AMBLE  ###################

  #Things to calculate beforehand

  #n: number runs in design

  n<-nrow(XQ)

  #nawp: number added WPs

  nawp<-nwp-nfwp

  #nx: number of quantitative factors

  nx<- ncol(XQ)

  #nf: number of runs in factorial points

  nf<- nfwp*nrpw

  #nv: number of +/- permutations

  nv<- choose(nrpw, nrpw/2)

  print("XQ")

  print(XQ)

###### PRE-AMBLE  ENDS ##################

############ METHOD 1: ADD SPLITPLOT #############

  #If method 1 is selected, use this algorithm

  addsp<-function()

  { print("Add SP function")
```

```
#Make matrix with all possible SP factor settings within a WP
vp<-rep(1, nrpw/2)
vn<-rep(-1, nrpw/2)
v<-c(vp,vn)

V<-perm(v)
V<-t(V)
print("V")
print(V)

#Create a matrix to keep track of allowable z's. Start with it empty.
#Also start count of "acceptable" z's

zero<-rep(0,n)
GoodZ<-matrix(data=zero, nrow=1, ncol=n)
countz<-0

#Create Matrix with all combos of within WP settings
combo<-1:nv
SPsettings<-permutations(n=nv,r=nfwp,v=combo, repeats.allowed=T)

#Create Candidate X matrix
z0<-rep(0,nrpw)

#Choose WP settings to use first
for(i in 1:nrow(SPsettings)){

  discard<-FALSE
```

```
countz<-countz+1

z<-z0
for(j in 1:(nawp-1)){
  z<-c(z,z0)
}
#Cycle through each WP and add to the candidate z
for(j in 1:nfwp)
{
  z<-c(V[,SPsettings[i,j]],z)
}

X<-matrix(data=c(XQ,z), nrow=n, ncol=(nx+1), byrow=FALSE)

#Make Paritioned Design

#Create subsetted matrices in the factorial points where z is plus and z is minus

zp<-vector(length=nfwp*nrpw*(nx+1)/2)
zm<-vector(length=nfwp*nrpw*(nx+1)/2)
ZP<-matrix(data=zp, nrow=nfwp*nrpw/2, ncol=(nx+1))
ZM<-matrix(data=zm, nrow=nfwp*nrpw/2, ncol=(nx+1))

#Counters for which row we are filling in the plus and minus matrices
rowp<-1
rowm<-1

#Sort into subsetted designs
```

```
for(a in 1:nf)
{
  if(X[a,(nx+1)]==-1)
  {
    ZM[rowm,]<-X[a,]
    rowm<-rowm+1
  }

  if(X[a,(nx+1)]==1)
  {
    ZP[rowp,]<-X[a,]
    rowp<-rowp+1
  }

}

#Delete categorical column from ZM and ZP
ZM<-ZM[,-(nx+1)]
ZP<-ZP[,-(nx+1)]

#Check for balance
for(a in 1:nx){
  if(discard==TRUE) break
  if(sum(ZM[,a])!=0){
    discard<-TRUE

  }
```

```
  if(sum(ZP[,a])!=0){
    discard<-TRUE

  }
}

#Check for orthogonality
#Check in subsetted matrices

if(discard==FALSE){

  for(a in 1:(nx-1)){
    for(b in (a+1):nx){
      if(discard==TRUE) break
      if(sum(ZM[,a]*ZM[,b])!=0){
        discard<-TRUE
      }

      if(sum(ZP[,a]*ZP[,b])!=0){
        discard<-TRUE

      }

    }
  }
}

#Keep if no red flags, else discard
```

```
  if(discard==FALSE){

    GoodZ<-rbind(GoodZ,z)
  }

}

#Delete place holder row of zeros
GoodZ<-GoodZ[-1,]
GoodZ<-GoodZ[,-c(nf+1:n)]
print("GoodZ")
print(GoodZ)

#Count number of good rows
print("Number of GoodZ")
print(nrow(GoodZ))

if(nrow(GoodZ)==0){
  print("No eligible designs")
}

if((nrow(GoodZ))!=0){

  #Set current Max D-criterion to 0
  maxD<-0

  #Set current max D-criterion z-row to be empty
  maxZ<-rep(0,n)
```

```
M<-matrix(0, nrow=n, ncol=nwp)

row<-1
for(k in 1:nwp){
  for(j in 1:nrpw){
    M[row,k]<-1
    row<-row+1
  }
}

M
In<-diag(1,n,n)

#Set count to zero to check counts
count<-0

#Take each GoodZ row, add in remaining z's
numZ<-nrow(GoodZ)

# Determine if range of vector is 0.
zero_range <- function(x, tol = .Machine$double.eps ^ 0.5) {
  if (length(x) == 1) return(TRUE)
  x <- range(x) / mean(x)
  isTRUE(all.equal(x[1], x[2], tolerance = tol))
}
Check<-rep(0,nawp)
```

```
#Check which added whole-plots need to cycle through possible wp assignment
for(i in 1:nawp)
{
  testzero<-nx
  for(j in 1:nx)
  {
    #If wp same, count down number diff
    if(zero_range(XQ[(nfwp*nrpw+(i-1)*nrpw+1):(nfwp*nrpw+(i-1)*nrpw+1+(nrpw-1)),j]
    {
      testzero<-testzero-1
    }
  }

  #If all columns were flagged as the same
  if(testzero!=0)
  {
    Check[i]<-1
  }

}
print("Check")
print(Check)

if(sum(Check)==0)
{
  SPAsettings=rep(1,nawp)
}
```

```
if(sum(Check)!=0)
{

  #Create Matrix with all combos of within WP settings for added WPs
  combo<-1:nv

  #Find all possibile permutations for the whole-plots that need to rotate
  OSPAsettings<-permutations(n=nv, r=sum(Check), v=combo, repeats.allowed=T)

  #Number of potential setting combinations
  nSPS=nrow(OSPAsettings)

  #Added columns of one's for all added wps that don't need to rotate
  one<-rep(1,nSPS)
  #Tracks which OSPsettings column needs to be added in next
  SPcol<-1

  SPAsettings=matrix(nrow=nSPS, ncol=nawp)

  for(i in 1:nawp)
  {
    #If added wholeplot doesn't need to rotate, assign one
    if(Check[i]==0)
    {
      SPAsettings[,i]=one
    }
```

```
    if(Check[i]==1)
    {
      SPAsettings[,i]=OSPAsettings[,SPcol]
      SPcol<-SPcol+1
    }
  }
}


print("Number of SPAsettings")
print(nrow(SPAsettings))


countgood<-0
countopt<-0

for(a in 1:numZ){
  for(b in 1:nrow(SPAsettings)){

    count<-count+1

    #Get z with added points
    z<-GoodZ[a,]

    for(c in 1:ncol(SPAsettings)){

      z<-c(z,V[,SPAsettings[b,c]])
```

```
}

X<-matrix(data=c(XQ,z), nrow=n, ncol=(nx+1), byrow=FALSE)

#Test for orthogonality
discard<-FALSE

for(d in 1:nx){
  for(e in (d+1):(nx+1)){

    if(sum(X[,d]*X[,e])!=0){
      discard<-TRUE
    }
  }
}

#If good, check for D-optimality

if(discard==FALSE){

  countgood<-countgood+1


  #Calculate D-criterion assuming equal variances
  D<-det(t(X)%*%solve((In+M%*%t(M)))%*%X)

  #Decide if better than max

  if(D==max(D)){
```

```
          countopt<-countopt+1
        }

        if(D>maxD){

          maxD<-D
          maxZ<-z
          countopt<-1

        }


      }
    }

  }

  print("maxZ")
  print(maxZ)
  print("maxD")
  print(maxD)
  print("countgood")
  print(countgood)
  print("countopt")
  print(countopt)

  #End of Goodz!=0 bracket
}
```

```
  }
######### METHOD 1: ADD SPLITPLOT ENDS #################



######### METHOD 2: ADD WP BY PSEUDO-PARTITIONING ######

  #If Method 2 is selected, use this algorithm

  ppwp<-function(){
    print("Pseudo-partitioning function")

    #Make vector of half 1/-1 to permute for WP assignments

    WPa<-rep(0,nfwp)
    for(i in 1:(nfwp/2)){
      WPa[i]<-1
    }

    for(i in (nfwp/2+1):nfwp){
      WPa[i]<- -1
    }


    #Create a matrix of all WP settings
    WPsettings<-perm(WPa)
    WPsettings<-t(WPsettings)
```

```
print("WPsettings for partitioning")
print(WPsettings)


#Create a matrix to keep track of allowable z's. Start with it empty.
#Also start count of "acceptable" z's

zero<-rep(0,n)
GoodZ<-matrix(data=zero, nrow=1, ncol=n)
countz<-0

#Create Candidate Matrix X

#z0 for added WPs we'll deal with later
z0<-rep(0,nrpw)

#Choose WP settings to use first
for(i in 1:ncol(WPsettings)){
  discard<-FALSE
  countz<-countz+1

  pone<-rep(1,nrpw)
  mone<-rep(-1,nrpw)

  z<-0
  #Cycle through each WP and add to the candidate z
  for(j in 1:nfwp){
```

```
  if(WPsettings[j,i]==-1){
    z<-c(z,mone)
  }

  if(WPsettings[j,i]==1){
    z<-c(z,pone)
  }

}

z<-z[-1]

for(k in 1:(nawp)){
  z<-c(z,z0)
}

X<-matrix(data=c(z,XQ), nrow=n, ncol=(nx+1),byrow=FALSE)

#Make Partitioned Design
#Create subsetted matrices in the factorial points where z is plus and z is minus

zp<-vector(length=nfwp*nrpw*(nx+1)/2)
zm<-vector(length=nfwp*nrpw*(nx+1)/2)
ZP<-matrix(data=zp, nrow=nfwp*nrpw/2, ncol=(nx+1))
ZM<-matrix(data=zm, nrow=nfwp*nrpw/2, ncol=(nx+1))

#Counters for which row we are filling in the plus and minus matrices
rowp<-1
```

```
rowm<-1

#Sort into subsetted matrices

for(a in 1:nf){

  if(X[a,1]==-1){
    ZM[rowm,]<-X[a,]
    rowm<-rowm+1
  }

  if(X[a,1]==1){
    ZP[rowp,]<-X[a,]
    rowp<-rowp+1
  }

}

#Delete categorical column from ZM and ZP
ZM<-ZM[,-1]
ZP<-ZP[,-1]

#Check for balance

for(a in 1:nx){
  if(discard==TRUE) break
  if(sum(ZM[,a])!=0){
    discard<-TRUE
```

```
  }

  if(sum(ZP[,a])!=0){
    discard<-TRUE
  }
}

#Check for orthogonality
#Check in subsetted matrices

if(discard==FALSE){

  for(a in 1:(nx-1)){
    for(b in (a+1):nx){
      if(discard==TRUE) break
      if(sum(ZM[,a]*ZM[,b])!=0){
        discard<-TRUE
      }

      if(sum(ZP[,a]*ZP[,b])!=0){
        discard<-TRUE
      }
    }
  }
}

#Keep is no red flags, else discard
if(discard==FALSE){
```

```
    GoodZ<-rbind(GoodZ,z)
  }

  #This is the end of num(WPsettings bracket)
}

#Delete place holder row of zeros
GoodZ<-GoodZ[-1,]
GoodZ<-GoodZ[,-c(nf+1:n)]
print("GoodZ")
print(GoodZ)

#Count number of good rows
print("Number of Goodz")
print(nrow(GoodZ))

if(nrow(GoodZ)==0){
  print("No eligible designs")
}

if(nrow(GoodZ)!=0){
  #Set current Max D-criterion to 0
  maxD<-0

  #Set current max D-criterion z-row to be empty
  maxZ<-rep(0,n)

  M<-matrix(0,nrow=(n+nawp*nrpw), ncol=(nwp+nawp))
```

```
row<-1
for(k in 1:(nwp+nawp)){
  for(j in 1:nrpw){
    M[row,k]<-1
    row<-row+1
  }
}

M
In<-diag(1,(n+nawp*nrpw),(n+nawp*nrpw))

#Set count to zero to check counts
count<-0

#Take each GoodZ row, add in remaining z's
numZ<-nrow(GoodZ)

#Add in extra WPs (these will eventually get all -1 WP setting)

for(i in 1:nawp){
  for(j in 1:nrpw){
    XQ<-rbind(XQ,XQ[(nf+nrpw*(i-1)+j),])
  }
}

countgood<-0
countopt<-0
```

```
for(a in 1:numZ){

  count<-count+1

  #Get z with added point
  z<-GoodZ[a,]

  for(b in 1:nawp){
    z<-c(z,pone)
  }

  for(b in 1:nawp){
    z<-c(z,mone)
  }

  X<-matrix(data=c(z,XQ), nrow=(n+nawp*nrpw), ncol=(nx+1), byrow=FALSE)

  #Test for orthogonality
  discard<-FALSE

  for(d in 1:nx){
    for(e in (d+1):(nx+1)){
      if(discard==TRUE) break
      if(sum(X[,d]*X[,e])!=0){
        discard<-TRUE
      }
    }
```

```
  }

  #If good, check for d-optimality

  if(discard==FALSE){
    countgood<-countgood+1


    #Calculate D-criterion assuming equal variances
    D<-det(t(X)%*%solve((In+M%*%t(M)))%*%X)

    #Decide if better than Max

    if(D==maxD){
      countopt<-countopt+1
    }

    if(D>maxD){

      maxD<-D
      maxZ<-z
      countopt<-1
    }


    #This is the end of the test d-optimality if "good" bracket
  }
  #this is the end of the cycle through GoodZ bracket
}
```

```
      print("maxZ")
      print(maxZ)
      print("maxD")
      print(maxD)
      print("countgood")
      print(countgood)
      print("countopt")
      print(countopt)


    }
    #this is the end of ppwp bracket
  }

######## METHOD 2: ADD WP BY PSEUDO-PARTITIONING ENDS ######

######## METHOD 3: SPLIT WHOLEPLOTS BEGINS #################
  #If method 3 is selected, use this algorithm.

  splitwp<-function(){

    #######
    print("Split WP Function")

    #Make matrix with all possible partitions of the WP.
    vp<-rep(1, nrpw/2)
    vn<-rep(-1, nrpw/2)
    v<-c(vp,vn)
```

```
V<-perm(v)
V<-t(V)
print("V")
print(V)

#Create a matric to keep track of allowable z's. Start with it empty.
#Also start count of "acceptable" z's

zero<-rep(0,n)
GoodZ<-matrix(data=zero, nrow=1, ncol=n)
countz<-0

#Create a matrix with all possible WP partition assignments from V matrix
combo<-1:nv
WPsettings<-permutations(n=nv, r=nfwp, v=combo, repeats.allowed=T)

#Create candidate X matrix
z0<-rep(0,nrpw)

#Choose WP assignments to use first

for(i in 1:nrow(WPsettings)){

  discard<-FALSE
  countz<-countz+1

  z<-z0
  for(j in 1:(nawp-1)){
```

```
  z<-c(z,z0)
}

#Cycle through each WP and add to the candidate z
for(j in 1:nfwp){
  z<-c(V[,WPsettings[i,j]],z)
}

X<-matrix(data=c(z,XQ), nrow=n, ncol=(nx+1), byrow=FALSE)

#Make Partitioned Design

#Create subsetted matrices in the factorial points where z is plus and z is minus

zp<-vector(length=nfwp*nrpw*(nx+1)/2)
zm<-vector(length=nfwp*nrpw*(nx+1)/2)
ZP<-matrix(data=zp, nrow=nfwp*nrpw/2, ncol=(nx+1))
ZM<-matrix(data=zm, nrow=nfwp*nrpw/2, ncol=(nx+1))

#Counters for which row we are filling in the plus and minus matrics
rowp<-1
rowm<-1

#Sort into subsetter designs

for(a in 1:nf){

  if(X[a,1]==-1){
```

```
    ZM[rowm,]<-X[a,]
    rowm<-rowm+1
  }

  if(X[a,1]==1){
    ZP[rowp,]<-X[a,]
    rowp<-rowp+1
  }
}

#Delete categorical column from ZM and ZP
ZM<-ZM[,-(1)]
ZP<-ZP[,-1]

#Check for balance in subsetted columns
for(a in 1:nx){
  if(discard==TRUE) break
  if(sum(ZM[,a])!=0){
    discard<-TRUE
  }

  if(sum(ZP[,a])!=0){
    discard<-TRUE
  }
}

#Check for orthogonality
#Check in subsetted matrices
```

```
if(discard==FALSE){

  for(a in 1:(nx-1)){
    for(b in (a+1):nx){
      if(discard==TRUE) break
      if(sum(ZM[,a]*ZM[,b])!=0){
        discard<-TRUE
      }
    }
  }
}

#Check subsetted matrices are balanced within whole-plots

if(discard==FALSE){

  for( i in (nwx+1):nx){
    for(j in 1:nfwp){
      if(discard==TRUE) break
      if(sum(ZP[(nrpw/2*(j-1)+1):(nrpw/2*(j-1)+1+(nrpw/2-1)),i])!=0){
        discard<-TRUE
      }
    }
  }

  for( i in (nwx+1):nx){
    for(j in 1:nfwp){
```

```
        if(discard==TRUE) break
        if(sum(ZM[(nrpw/2*(j-1)+1):(nrpw/2*(j-1)+1+(nrpw/2-1)),i])!=0){
          discard<-TRUE
        }
      }
    }
  }

  #Keep if no red flags, else discard
  if(discard==FALSE){
    GoodZ<-rbind(GoodZ,z)
  }
}

#Delete place holder row of zeros
GoodZ<-GoodZ[-1,]
GoodZ<-GoodZ[,-c(nf+1:n)]
print("GoodZ")
print(GoodZ)

#Count number of good rows
print("Number of GoodZ")
print(nrow(GoodZ))

if(nrow(GoodZ)==0){
  print("No eligible designs")
}
```

```
if(nrow(GoodZ)!=0){

  #Set current Max D-criterion to 0
  maxD<-0

  #Set current max D-criterion z-row to be empty
  maxZ<-rep(0,n)

  M<-matrix(0,nrow=n, ncol=nwp*2)

  row<-1
  for(k in 1:(nwp*2)){
    for(j in 1:(nrpw/2)){

      M[row,k]<-1
      row<-row+1
    }

  }

  M
  In<-diag(1,n,n)

  #Set count to zero to check counts
  count<-0

  #Take each GoodZ row, add in remaining z's
  numZ<-nrow(GoodZ)
```

```
# Determine if range of vector is 0.
zero_range <- function(x, tol = .Machine$double.eps ^ 0.5) {
  if (length(x) == 1) return(TRUE)
  x <- range(x) / mean(x)
  isTRUE(all.equal(x[1], x[2], tolerance = tol))
}
Check<-rep(0,nawp)

#Check which added whole-plots need to cycle through possible wp assignment
for(i in 1:nawp)
{
  testzero<-nx
  for(j in 1:nx)
  {
    #If wp same, count down number diff
    if(zero_range(XQ[(nfwp*nrpw+(i-1)*nrpw+1):(nfwp*nrpw+(i-1)*nrpw+1+(nrpw-1)),j]
    {
      testzero<-testzero-1
    }
    # print("testzero")
    # print(testzero)
  }

  #If all columns were flagged as the same
  if(testzero!=0)
  {
    Check[i]<-1
```

```
  }

}
print("Check")
print(Check)

if(sum(Check)==0)
{
  WPAsettings=rep(1,nawp)
}

if(sum(Check)!=0)
{


  #Create Matrix with all combos of within WP settings for added WPs
  combo<-1:nv

  #Find all possibile permutations for the whole-plots that need to rotate
  OWPAsettings<-permutations(n=nv, r=sum(Check), v=combo, repeats.allowed=T)

  #Number of potential setting combinations
  nWPS=nrow(OWPAsettings)

  #Added columns of one's for all added wps that don't need to rotate
  one<-rep(1,nWPS)
  #Tracks which OWPsettings column needs to be added in next
  WPcol<-1
```

```
  WPAsettings=matrix(nrow=nWPS, ncol=nawp)

  for(i in 1:nawp)
  {
    #If added wholeplot doesn't need to rotate, assign one
    if(Check[i]==0)
    {
      WPAsettings[,i]=one
    }

    if(Check[i]==1)
    {
      WPAsettings[,i]=OWPAsettings[,WPcol]
      WPcol<-WPcol+1
    }
  }
}

print("Number WPAsetting")
print(nrow(WPAsettings))
countgood<-0
countopt<-0

for(a in 1:numZ){
  for(b in 1:nrow(WPAsettings)){
    count<-count+1
```

```
#Get z with added points
z<-GoodZ[a,]

for(c in 1:ncol(WPAsettings)){
  z<-c(z,V[,WPAsettings[b,c]])
}

X<-matrix(data=c(z,XQ), nrow=n, ncol=(nx+1), byrow=FALSE)

#Take X, divide factorial point into XP and XQ, put them one on top of the
#other and replace the factorial points. Works because of order we sort
#through original matrix from top to bottom

#Create subsetted matrices in the factorial points where z is plus and z is mi

zp<-vector(length=nfwp*nrpw*(nx+1)/2)
zm<-vector(length=nfwp*nrpw*(nx+1)/2)
ZP<-matrix(data=zp, nrow=nfwp*nrpw/2, ncol=(nx+1))
ZM<-matrix(data=zm, nrow=nfwp*nrpw/2, ncol=(nx+1))

#Counters for which row we are filling in the plus and minus matrices
rowp<-1
rowm<-1

#Sort into subsetted designs

for(e in 1:nf){
```

```
  if(X[e,1]==-1){
    ZM[rowm,]<-X[e,]
    rowm<-rowm+1
  }

  if(X[e,1]==1){
    ZP[rowp,]<-X[e,]
    rowp<-rowp+1
  }
}

X<-X[-(1:nf),]
X<-rbind(ZP,ZM,X)

#Test for orthogonality
discard<-FALSE

for(d in 1:nx){
  for (e in (d+1):(nx+1)){
    if(discard==TRUE) break
    if(sum(X[,d]*X[,e])!=0){
      discard<-TRUE
    }
  }
}

#If good, check for D-optimality
if(discard==FALSE){
```

```
      countgood<-countgood+1


      #Calculate D-criterion assuming equal variances
      D<-det(t(X)%*%solve((In+M%*%t(M)))%*%X)

      #Decide if better than max

      if(D==maxD){
        countopt<-countopt+1
      }

      if(D>maxD){

        maxD<-D
        maxZ<-z
        countopt<-1
      }


    }
    #This bracket ends cycling through all potential designs
  }
}

print("maxZ")
print(maxZ)
print("maxD")
```

```
      print(maxD)

      print("countgood")

      print(countgood)

      print("countopt")

      print(countopt)


      #This bracket ends the "if non-empty goodZ cyle"

    }

    #This brackets ends method 3

  }

###### METHOD 3: SPlIT WHOLEPLOTS ENDS#####################

  #If method 4 is selected, use this algorithm

  doublewp<-function(){

    #Create a matrix to keep track of allowable z's. Start with it empty

    #Also start count of "acceptable" z's

    zero<-rep(0,n*2)

    GoodZ<-matrix(data=zero, nrow=1, ncol=nwp*2)

    #Double the size of XQ

    #This bracket ends the double wp function

  }

###### METHOD 4: ADD WP BY DOUBLING DESIGN ###############

  #If method 4 is selected, use this algorithm.
```

```
doublewp<-function(){

  print("Double WP Function")

  #Double matrix
  XQ<-rbind(XQ, XQ)



  #Add in blank WP column
  z<-rep(0,(2*n))

  #print(z)
  XQ<-cbind(z,XQ)



  #Add in +/- 1's

  for(i in 1:n){
    XQ[i,1]<- -1
    XQ[(n+i),1]<-1
  }

  X<-XQ



  #Calculate D-criterion assuming equal variances
```

```
    M<-matrix(0,nrow=(2*n), ncol=nwp*2)

    row<-1
    for(k in 1:(nwp*2)){
      for(j in 1:(nrpw)){

        M[row,k]<-1
        row<-row+1
      }

    }

    #M
    In<-diag(1,(2*n),(2*n))

    D<-det(t(X)%*%solve((In+M%*%t(M)))%*%X)

    print("X")
    print(X)
    print("D-criterion")
    print(D)


    #This bracket ends the double wp function
  }
######## METHOD 4: ADD WP BY DOUBLING DESIGN ENDS ##############
  if(method==1)
  { addsp() }
```

```
  if(method==2)
  { ppwp() }

  if(method==3)
  {splitwp()}

  if(method==4)
  {doublewp()}


}
```

Bibliography

Aggarwal, M. L. and Bansal, A. (1998) "Robust Response Surface Design for Quantitative and Qualitative Factors". *Communications in Statistics - Theory and Methods* 27, pp. 89-106.

Aggarwal, M.L.; Gupta B.C.; and Bansal, A. (2000) "Small Robust Response Surface Designs for Quantitative and Qualitative Factors". *American Journal of Mathematical and Management Sciences* 39, pp. 103-130.

Allen, T. T. and Tseng, S.-H. (2011) "Variance Plus Bias Optimal Response Surface Designs with Qualitative Factors Applied to Stem Choice Modeling". *Quality and Reliability Engineering International* 27, pp. 1119-1210.

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

Bingham, D. and Sitter, R.R. (2001). "Design Issues in Fractional Factorial Split-Plot Experiments". *Journal of Quality Technology* 33, pp. 2-15.

Bingham, D. and Sitter, R. (2003) "Fractional Factorial Split-Plot Designs for Robust Parameter Experiments". *Technometrics* 45, pp. 80-89

Brenneman, W.A and Myers, W.R. (2003) “Robust Parameter Design With Categorical Noise Variables”. *Journal of Quality Technology.* 35, pp. 335-341.

Chantarat, N.; Zhend, N.; Allen, T.; and Huang, D. (2003) “Optimal Experimental Design For Systems Involving Both Quantitative and Qualitative Factors” *Proceedings of the 2003 Winter Simulation Conference*

Cheng, S.W. and Wu, C.F.J. (2001). “Factor Screening and Response Surface Exploration”. *Statistica Sinica* 11, pp. 533-604.

Cook, R. D. and Nachtsheim, C. J. (1989) “Computer-Aided Blocking of Factorial and Response-Surface Designs”. *Techometrics* 31, pp. 339-346.

Draper, N. R. and John, J. A. (1988). “Response Surface Designs For Quantitative and Qualitative Variable”. *Technometrics* 30, pp. 423-428.

Draper, N. R. and John, J. A. (1998). “Response Surface Designs Where Levels of Some Factors Are Difficult to Change”. *Australian and New Zealand Journal of Statistics* 40, pp. 487-495.

Fisher, R.A. (1925). *Statistical Methods for Research Workers.* Edinburgh, UK: Oliver and Boyd.

Georgiou, S.D.; Stylianou, S.; and Aggarwal, M. (2014). “Efficient Three-Level Screening Designs Using Weighing Matrices”. *Statistics* 48(4), pp. 815-833.

Gilmour, S. G. (2006). "Response Surface Designs for Experiments in Bioprocessing". *Biometrics* 62, pp. 323-331.

Goos, P. and Vanderbroek, M. (2001). "Optimal Split-Plot Designs". *Journal of Quality Technology* 33, pp. 436-450.

Goos, P. (2002). *The Optimal Design of Blocked and Split-Plot Experiments.* New York: Springer.

Goos, P. (2006). "Optimal Versus Orthogonal and Equivalent Estimation Design of Blocked and Split-Plot Experiments". *Statistica Neerlandica* 60, pp. 361-378.

Hadj-Alounane, A. and Bean, J. (1997). "A Genetic Algorithm For The Multiple-Choice Integer Program". *Operations Research* 45, pp. 92-101.

Huang, P., Chen, D., and Voelkel, J. (1998), "Minimum Aberration Two-Level Split-Plot Designs". *Technometrics* 40, pp. 314-326.

Huang, M.-N. L.; Lee, C.-P.; Chen, R.-B.; and Klein, T. (2010). "Exact *D*-Optimal Designs for a Second-Order Response Surface Model on a Circle with Qualitative Factors". *Computational Statistics and Data Analysis* 54, pp. 516-530.

Jones, G. and Goos, P. (2012a). "An Algorithm for Finding D-Efficient Equivalent-Estimation Second-Order Split-Plot Designs". *Journal of Quality and Technology* 44, pp. 363-374.

Jones, G. and Goos, P. (2012b). "I-Optimal Versus D-Optimal Split-Plot Response Surface Designs". *Journal of Quality Technology* 44, pp. 85-101.

Jones, B. and Nachtscheim, C. J. (2009). "Split-Plot Designs: What, Why and How". *Journal of Quality Technology* 41, pp. 340-361.

Jones, B. and Nachtscheim, C. J. (2011a). "Efficient Designs With Minimal Aliasing". *Technometrics* 531, pp. 62-71.

Jones, B. and Nachtscheim, C. J. (2011b). "A Class of Three-Level Designs for Definitive Screening in the Presence of Second-Order Effects". *Journal of Quality Technology* 43, pp. 1-15.

Jones, G. and Nachtscheim, C.J. (2013), "Definitive Screening Designs with Added Two-Level Categorical Factors". *Journal of Quality and Technology* 45, pp. 121-129.

Jourdan, A. and Zabalza-Mezghani, I. (2004). "Response Surface Designs for Scenario Management and Uncertainty Quantification in Resevoir Production". *Mathematical Geology* 36, pp. 965-985.

Katsaounis, T. I. , (2008). "Equivalence of Factorial Design 965-s With Qualitative And Quantitative Factors". *Journal of Statistical Planning and Inference* 142, pp. 79-85.

Kowalski, S. M.; Cornell, J.A.; and Vining, G.G., (2002). "Split-Plot Designs and Estimation Methods for Mixture Experiments With Process Variables". *Technometrics* 44, pp. 72-79.

Kowalski, S. M.; Borror, C.M.; and Montgomery, D. C., (2005) "A Modified Path of Steepest Ascent for Split-Plot Experiments". *Journal and Quality Technology* 37, pp. 75-83.

Lee, C-P. and Huang, M-N. L. (2011) "D-Optimal Designs for Second-Order Response Surface Models with Qualitative Factors". *Journal of Data Science* 9, pp. 139-153.

Lekivetz, R. and Tang, B. (2011). "Robust Designs Through Partially Clear Two-Factor Interactions". *Biometrika* 98, pp. 733-739.

Letsinger, J.D.; Myers, R.H.; and Lenter, M. (1996). "Response Surface Methods for Bi-Randomization Structures". *Journal of Quality Technology* 28, pp. 381-397.

Lin, C-Y. (2015). "Construction and Selection of the Optimal Balanced Blocked Definitive Screening Design". *Metric* 78, pp. 373-383.

Loeppky, J.L. and Sitter, R. (2002) "Analyzing Unreplicated Blocked or Split-Plot Fractional Factorial Designs". *Journal of Quality Technology* 34, pp. 229. Lucas, J. M. (1976). "Which Response Surface Design is Best: A Performance Comparison of Several Types of Quadratic Response Surface Designs in Symmetric Regions". *American Society for Quality* 18, pp. 411-417.

Macharia, H. and Goos, P. (2010). "D-Optimal and D-Efficient Equivalent Estimation Second-Order Split-Plot Designs". *Journal of Quality and Technology* 42, pp. 358-372.

Mee, R.W. (2007). “Optimal Three-Level Designs for Response Surfaces in Spherical Experimental Regions”. *Journal of Quality Technology* 39, pp. 340-354.

McLeod, R.G. and Brewster, J.F. (2004). “The Design of Blocked Fractional Factorial Split-Plot Experiments” *Technometrics* 46, pp. 135-146.

Montogomery, D. C. (2012). *Design and Analysis of Experiments, 8th Edition.* New York, NY: John Wiley.

Myers, R.H. and Montgomery, D.C. (1995). *Response Surface Methodology: Process and Product Optimization Using Designed Experiments.* New York, NY: John Wiley.

Myers, R.H.; Montgomery, D. C.; Vining, G. G; Borror, C. M.; and Kowalski, S.M. (2004). “Response Surface Methodology: A Retrospective and Literature Survery”. *Journal of Quality Technology* 2004, pp. 53-77.

Nguyen, N-K, Pham, T-D. (2015). “Searching for D-Effiicient Equivalent-Estimation Second-Order Split-Plot Designs”. *Journal of Quality Technology* 47, pp. 54-66.

Parker, P. A.; Kowalski, S. M.; and Vining, G.G. (2007a). “Construction of Balanced Equivalent Estimation Second-Order Split-Plot Designs”. *Technometrics* 49, pp. 56-65.

Parker, P. A.; Kowalski, S. M.; and Vining , G.G. (2007b). “Unbalanced and Minimal Point Equivalent Estimation Second-Order Split-Plot Designs”. *Journal of Quality Technology* 39, pp. 376-388.

Qian, P. and Wu, C.F.J. (2009). "Slice Space-Filling Designs". *Biometrika* 96, pp. 945-956.

Qian, P.Z.G.; Wu, Q.; and Wu, C.F.J. (2008). "Gaussian Process Models for Computer Experiments With Qualitative and Quantitative Factors". *Technometrics* 50, pp. 383-396.

Ranjan, P.; Bingham, D.; and Mukerjee, R. (2010). "Stars and Regular Fractional Factorial Designs with Randomization Restrictions". *Statistica Sinica* 20, pp. 1637-1653.

Robinson, T.J.; Borror, C.M.; and Myers, R.H. (2004). "Robust Parameter Design: A Review". *Quality and Reliability Engineering International* 20, pp. 81-101.

Sarika (2005). "Response Surface Designs for Quantitative and Qualitative Factors" (Unpublished doctoral thesis). I.A.S.R.I., New Delhi.

Schwabe, R. and Wierich, W. (1995). "D-Optimal Designs of Experiments With Non-Interacting Factors". *Journal of Statistical Planning and Inference* 44, pp. 371-384.

Trinca, L.A. and Gilmour, S.G. (2001). "Multistratum Response Surface Designs". *Technometrics* 43, pp. 25-33.

Victorbabu, B.R. and Vasundharadevi, V. (2009) "On Modified Second Order Response Surface Designs, Rotatable Designs, Rotatable Design With Equi-Spaced Doses". *International Journal of Agricultural and Statistics Sciences* 5, pp. 425-435

Vining, G.G. and Kowalski, S.M. (2008). "Exact Inference for Response Surface Designs Within a Split-Plot Structure". *Journal of Quality Technology* 40, pp. 394-406.

Vining, G.G.; Kowalski, S.M.; and Montgomery, D.C. (2005). "Response Surface Designs Within a Split-Plot Structure". *Journal of Quality Technology* 37, pp. 115-129.

Wang, L. (2006). "Recommendations for Design Parameters for Central Composite Designs with Restricted Randomization". Virginia Polytechnic Institute and University, Blacksburg, Virginia.

Wang, L.; Kowalski, S.M.; and Vining, G.G. (2009). "Orthogonal Blocking of Response Surface Split-Plot Designs". *Journal of Applied Statistics* 36, pp. 303-321.

Wang, L.; Vining G.G.; and Kowalski, S.M. (2010). "Two-Strata Rotatability in Split-Plot Central Composite Designs". *Applied Stochastic Models in Business and Industry* 26, pp. 431-447.

Wu, C.F.J. and Ding, Y. (1998). "Construction of Response Surface Designs for Qualitative and Quantitative Factors". *Journal of Statistical Planning and Inference* 71, pp. 331-348.

Xu, X.; Haaland, B.; and Qian, P.Z.G. (2011). "Sudoku-Based Space-Filling Designs". *Biometrika* 98, pp. 711-720.

Xiao, L; Line, D.K.J.; and Bai, F. (2012). "Constructing Definitive Screening Designs Using Conference Matrices". *Journal of Quality Technology* 44, pp. 2-8.

Yang, J.; Mandal, A.; and Majumdar, D. (2012) "Designs For Two-Level Factorial Experiments With Binary Response". *Statistica Sinica* 22, pp. 885-907.