2012

Applications of Sure Independence Screening Analysis for Supersaturated Designs

Lindsey Nicely
Virginia Commonwealth University

Applications of Sure Independence Screening Analysis for Supersaturated Designs

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Mathematical Sciences with Statistics concentration at Virginia Commonwealth University.

by
Lindsey B. Nicely
B.S. Mathematics, Furman University, 2004

Advisor: David J. Edwards, Ph.D.
Assistant Professor of Statistics
Department of Statistical Sciences and Operations Research

Virginia Commonwealth University
Richmond, Virginia
May 2012

Applications of Sure Independence Screening Analysis for Supersaturated Designs

by Lindsey B. Nicely
B.S. Mathematics, Furman University, 2004

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Mathematical Sciences with Statistics concentration at Virginia Commonwealth University.

Virginia Commonwealth University, May 2012

Advisor: David J. Edwards, Ph.D.
Assistant Professor of Statistics
Department of Statistical Sciences and Operations Research

(ABSTRACT)

Experimental design has applications in many fields, from medicine to manufacturing. Incorporating statistics into both the planning and analysis stages of the experiment will ensure that appropriate data are collected to allow for meaningful analysis and interpretation of the results. If the number of factors of interest is very large, or if the experimental runs are very expensive, then a supersaturated design (SSD) can be used for factor screening. These designs have n runs and $k > n - 1$ factors, so there are not enough degrees of freedom to allow estimation of all of the main effects.

This paper will first review some of the current techniques for the construction and analysis of SSDs, as well as the analysis challenges inherent to SSDs. Analysis techniques of Sure Independence Screening (SIS) and Iterative Sure Independence Screening (ISIS) are discussed, and their applications for SSDs are explored using simulation, in combination with the Smoothly Clipped Absolute Deviation (SCAD) approach for down-selecting and estimating the effects.

Dedication

This thesis is dedicated to my husband, Ben,
to my parents, Lee and Joe, and to my sister, Lauren,
for all of their love and support throughout this journey.
Also to my dogs, Copper and Cammi, for their unconditional
love, and for reminding me that every once in a while,
you just need to go out and play.

Acknowledgments

I would like to acknowledge and give special thanks to my advisor, Dr. David Edwards, for all of his time and patience spent helping me through this process. Thank you for the many hours you spent reviewing code and text, and for all of your encouragement along the way.

I also thank Dr. D'Arcy Mays and Dr. Kellie Archer for serving on my thesis committee, and for your careful review and advise.

Thank you to my professors and fellow grad students in the SS/OR department at VCU for keeping things fun, keeping me sane, and for all of your support for the past three years.

Finally, I would like to thank MeadWestvaco and my colleagues there for supporting my desire to go back to school, and for all of their support and understanding throughout my degree program.

Contents

1 Introduction **1**

2 Literature Review **7**

2.1 Introduction and Background 7

2.2 Design Construction Methods 8

2.2.1 Search Techniques 8

2.2.2 Construction from Existing Orthogonal Design Structure 11

2.3 Ranking Criteria 13

2.4 Analysis Methods 15

3 The Sure Independence Screening Approach **20**

3.1 Penalized Least Squares and the Smoothly Clipped Absolute Deviation (SCAD) Penalty 20

3.2 Sure Independence Screening (SIS) 22

3.3 Iterative Sure Independence Screening (ISIS) 23

4 Simulation Studies **24**

4.1 Williams Rubber Data 24

4.1.1 Comparison of SCAD and Stepwise Variable Selection Procedures by Li and Lin (2002) 24

4.1.2 Comparison of SCAD, Stepwise and Permutation Variable Selection Procedures by Koh, Eskridge and Wang (2011) 26

4.1.3 Comparison of SIS and ISIS Variable Selection Procedures 28

4.1.4 Application of SIS and ISIS Variable Selection Procedures to the Actual Williams Rubber Data 31

4.2 Marley and Woods Designs 34

4.2.1 Comparison of Forward Selection, Gauss-Dantzig Selector, and Model Averaging Variable Selection Procedures by Marley and Woods (2010) 34

4.2.2 Comparison of SIS and ISIS Variable Selection Procedures 39

5 Conclusion **50**

5.1 Concluding Remarks 50

5.2 Further Research 51

A R Code - Williams Rubber Data Design Simulations **55**

B R Code - Williams Rubber Data with Original Response **66**

C R Code - Marley and Woods Designs **69**

List of Tables

1.1 The 2^2 full factorial design 2

1.2 The estimable effects for an unreplicated 2^5 full factorial design 3

1.3 The 2^{3-1} fractional factorial design with defining relation $I = ABC$ 4

2.1 A small supersaturated design in $k = 10$ factors with $n = 6$ runs 8

2.2 The Plackett-Burman model matrix with $k = 11$ factors and $n = 12$ runs (an order $N = 12$ Hadamard matrix) 12

2.3 A supersaturated design with $k = 10$ factors and $n = 6$ runs (formed from a half-fraction of an $N = 12$ Hadamard matrix) 13

4.1 Half-fraction of the Williams rubber data 28-run Plackett-Burman design, as suggested by Lin (1993), with duplicate factor 16 removed (and columns 17 - 24 renumbered) 25

4.2 Original results of Li and Lin (2002) simulated comparison of SCAD and stepwise variable selection methods 26

4.3 Original results of Koh, Eskridge and Wang (2011) simulated comparison of SCAD, stepwise and permutation test variable selection methods with $N(0, 1)$ error distribution 27

4.4 Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods 29

4.5 Half-fraction of the Williams rubber data 28-run Plackett-Burman design, as suggested by Lin (1993), including the original response 31

4.6 Models recommended from analysis of the Williams (1968) rubber data . . . 32

4.7 Models recommended by SIS-SCAD and ISIS-SCAD for the half-fraction Williams (1968) rubber data set 33

4.8 Marley and Woods (2010) Bayesian D-optimal design with 22 factors and 18 runs 35

4.9 Marley and Woods (2010) Bayesian D-optimal design with 24 factors and 14 runs 35

4.10 Marley and Woods (2010) Bayesian D-optimal design with 26 factors and 12 runs 36

4.11 Marley and Woods (2010) $E(s^2)$-optimal design with 22 factors and 18 runs 36

4.12 Marley and Woods (2010) $E(s^2)$-optimal design with 24 factors and 14 runs 37

4.13 Marley and Woods (2010) $E(s^2)$-optimal design with 26 factors and 12 runs 37

4.14 Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the Bayesian D-optimal design in 22 factors and 18 runs 43

4.15 Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the Bayesian D-optimal design in 24 factors and 14 runs 44

4.16 Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the Bayesian D-optimal design in 26 factors and 12 runs 45

4.17 Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the $E(s^2)$-optimal design in 22 factors and 18 runs 46

4.18 Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the $E(s^2)$-optimal design in 24 factors and 14 runs 47

4.19 Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the $E(s^2)$-optimal design in 26 factors and 12 runs 48

4.20 Power results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for all six Marley and Woods (2010) designs, scenario 5 only (large number of effects of mixed size) 49

Chapter 1

Introduction

Experimental design has applications in many fields, from medicine to manufacturing. The goal of an experiment may be to determine which explanatory (or independent) variables are most influential to a particular response (or dependent variable), or to identify the optimal conditions for operating a process. In order to draw meaningful conclusions from the experiment, techniques of statistical design and analysis can be used. Incorporating statistics into both the planning and analysis stages of the experiment will ensure that appropriate data are collected to allow for meaningful analysis and interpretation of the results. Montgomery (2009) outlines the following steps for statistical experimental design:

1. Recognition of and statement of the problem.
2. Selection of the response variable.
3. Choice of factors, levels, and ranges.
4. Choice of experimental design.
5. Performing the experiment.
6. Statistical analysis of the data.
7. Conclusions and recommendations.

A **factor** is something that the experimenter wishes to vary during the experiment in order to determine its magnitude of influence over the response variable. As part of the statistical design process, specific **factor levels** are chosen for each factor. Often, two levels are selected for each factor – a low level and a high level. A benefit to choosing two levels vs. three or

more is that each factor effect will only use one degree of freedom. Designs such as this are referred to as **two-level designs**, and they will be the focus of this paper.

For a given experiment, the natural measurement units will likely vary from factor to factor (e.g. units of time, length, speed, temperature, etc.), and the numerical settings for the factors will typically vary greatly in range (e.g. temperature settings of 300°F and 400°F vs. the weights of an ingredient of 0.1 mg and 0.2 mg). Therefore, **coded levels** are assigned to the low and high levels for each factor, -1 and $+1$. Coding the factor levels allows for all of the factor effects to be compared on an even scale. The collection of coded factor levels can then be used to generate a two-level or $\mathbf{2^k}$ **factorial design**. The notation simply indicates that there are two levels for each of k factors. (Note that it is possible to create a three-level or 3^k factorial design with coded levels -1, 0 and $+1$, or any other design with other numbers of factor levels; however, those designs will not be discussed in this paper.)

The simplest example of a 2^k factorial design is that with two factors at two levels each, called the 2^2 factorial design, which is shown in Table 1.1. Here the notation for the coded levels, -1 and $+1$, has been simplified to $-$ and $+$. This is another commonly used method for denoting the low and high levels for a factor. In the tabular design notation used below, each column represents a single factor, and each row represents a **treatment combination**, the combination of factor settings that can be used for an experimental run. This 2^2 design can also be referred to as a 2^2 **full factorial design**, indicating that every possible treatment combination has been used in the design. Further, note that the design columns are orthogonal, meaning that $c_i' c_j = 0$ for the two design columns c_i and c_j.

Table 1.1: The 2^2 full factorial design

Factor A	Factor B
$-$	$-$
$-$	$+$
$+$	$-$
$+$	$+$

For the 2^2 factorial design, there are $2^2 = 4$ different treatment combinations. In a small design like this, **replication** might also be used, where each treatment combination would be performed two or more times. This provides additional degrees of freedom for the estimation of the error variance σ^2 and gives us extra confidence that we have an accurate representation of the true variation in response from varying the selected factors. Typically, the same number of replicates are used for each treatment combination. An important related aspect of statistical experimental design is **randomization**. For any design (with or without replicates), all runs should be performed in random order to prevent any systematic error from affecting the outcome.

Running a full factorial design will allow us to estimate all of the **main effects**, the direct effect of a single factor on the response, as well as all **interactions**, the combined simultaneous effect of two or more factors on the response. For example, running an unreplicated 2^2 factorial design gives us three degrees of freedom to estimate the effects of factor A, factor B, and the effect of the $A \times B$ interaction. A regression model representation of this simple 2^2 factorial design is given by

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2 + \epsilon, \tag{1.1}$$

where y is the response variable, β_0, β_1, β_2 and β_{12} are unknown parameters, x_1 and x_2 are the coded levels of factors A and B, respectively, and ϵ is an error term of normal distribution with mean $E(\epsilon) = 0$ and variance $Var(\epsilon) = \sigma^2$. Note, however, that we cannot estimate $Var(\epsilon)$ in an unreplicated 2^2 design with the interaction term, because there would not be enough degrees of freedom to do so. This is called a **saturated** design.

For this small example, it is probably reasonable to assume that we want to and have the resources to estimate both of the main effects and the interaction effect. However, even for an unreplicated 2^k factorial design, the number of required runs can increase quickly as the number of factors, k, increases. For example, a 2^3 factorial design will require eight runs, a 2^4 factorial design will require sixteen runs, and a 2^5 factorial design will require thirty-two runs. Often, the experimenter may be limited in terms of budget, time, or other resources, which will prevent all of the required runs for a full factorial design from being performed. For example, the 2^5 full factorial design with factors A, B, C, D and E would provide us with thirty-one degrees of freedom to estimate the main effects and all interactions, as outlined in Table 1.2.

Table 1.2: The estimable effects for an unreplicated 2^5 full factorial design

Main Effects	2-Factor Interactions	3-Factor Interactions	4-Factor Interactions	5-Factor Interaction
A, B, C, D, E	AB, AC, AD, AE, BC, BD, BE, CD, CE, DE	ABC, ABD, ABE, ACD, ACE, ADE, BCD, BCE, BDE, CDE	$ABCD$, $ABCE$ $ABDE$, $ACDE$, $BCDE$	$ABCDE$

An important and widely accepted principal of experimental design is **effect sparsity**, the assumption that only a few main effects and low-order interactions are usually important. For the 2^5 example, this means we may not need or want to estimate the 4-factor or 5-factor interaction effects (or maybe even some or all of the 3-factor interactions). Therefore, we can

use a smaller design to estimate just the main effects and lower-order interactions, rather than using the resource-intensive full factorial design.

A **fractional factorial design** can be used to perform a subset of the full factorial design runs. For example, an unreplicated 2^3 factorial design would require eight runs, but we can perform a **one-half fraction** of the 2^3 design in just four runs. The notation for this design is 2^{3-1}, which indicates that we started with a 2^3 factorial design and are performing a $2^{-1} = 1/2$ fraction. In general, the notation for a fractional factorial design is 2^{k-p}, where k is the number of factors, and we are taking the $(1/2)^p$ fraction of the full 2^k factorial design.

A 2^{k-p} fractional factorial design can be constructed using p design **generators**. We lose the ability to estimate the effects from any interaction terms selected as generators, so typically the generators are chosen from among the higher-order interaction terms. Recall that we assume the system is dominated by the main effects and lower-order interactions, so we do not need to be able to estimate all of the higher-order effects. The generators are selected and set equal to the identity column of 1's, I. This means that any interaction between generators will also be equivalent to I. Setting the generators equal to each other, to any interactions between generators, and to I determines the **defining relation** for the fractional factorial design. The terms in the defining relation (aside from I) are referred to as **words**. To form the 2^{k-p} fractional factorial design, we begin with a full factorial design in $k-p$ factors, and then the levels for the remaining factors are calculated from the design generators.

For example, suppose we want to create a one-half fraction of the 2^3 factorial design with factors A, B and C (i.e. a 2^{3-1} fractional factorial design). For this design, we would typically use $C = AB$ as the generator, and therefore the defining relation is $I = ABC$. To form the design, we begin with a full factorial in the first $3-1=2$ factors (i.e. a 2^2 factorial design in factors A and B). The coded levels for factor C are then calculated by multiplying the coded levels of factors A and B. Table 1.3 shows this 2^{3-1} fractional factorial design.

Table 1.3: The 2^{3-1} fractional factorial design with defining relation $I = ABC$

Factorial Effects		
A	B	$C = AB$
$-$	$-$	$+$
$-$	$+$	$-$
$+$	$-$	$-$
$+$	$+$	$+$

Note once again that each pair of columns in this one-half fractional factorial design are orthogonal. This unreplicated design has four experimental runs, and therefore three degrees of freedom that we can use to estimate the three main effects A, B and C, but none of the

interaction terms. On p.311 of his text, Montgomery (2009) has a table of recommended generators that can be used to create other 2^{k-p} designs for up to 15 factors.

The implication of using a fractional factorial design is that it results in **aliasing**. By multiplying various effects against the defining relation, we find that certain interactions are statistically indistinguishable from other main effects or interactions. For our 2^{3-1} example, the alias structure is

$$\begin{aligned} I &= ABC \\ A &= BC \\ B &= AC \\ C &= AB. \end{aligned} \tag{1.2}$$

Because we used $C = AB$ for the generator, ABC is aliased with the identity column I and is not estimable. Also, the three main effects are aliased with the three two-factor interactions, so we will not be able to distinguish which effect (the main effect or the corresponding two-factor interaction) is truly significant. Because of the **effect hierarchy** principal, we would typically assume that the main effect is more important than the associated two-factor interaction; however, that may not always be true, and we cannot tell from the analysis which is more important.

For any fractional factorial design, the shortest length word in the defining relation determines the **resolution** of the design. For example, our 2^{3-1} design has defining relation $I = ABC$, where the shortest word length is three. (We do not consider the length of I when determining the shortest word length.) Therefore, this is a **resolution III design**. There are specific aliasing patterns for designs of resolution III, IV and V. Resolution III designs have no main effects aliased with any other main effects, but some or all of the main effects will be aliased with two-factor interactions, and also some two-factor interactions will be aliased with each other. Resolution IV designs have no main effects aliased with any other main effects or two-factor interactions; however, some two-factor interactions will be aliased with each other. Finally, resolution V designs have no main effects or two-factor interactions aliased with any other main effects or two-factor interactions (Montgomery, 2009). To enable the estimation of as many main effects and low-order interactions as possible, we want the resolution of the design to be as high as possible. This is one reason that we typically look to the higher-order interaction terms when picking the design generators.

While fractional factorial designs do not allow for estimation of every interaction effect, they are usually designed to allow estimation of at least the main effects. Note that k degrees of freedom are required to estimate k main effects, which means such a design must have at least $k + 1$ experimental runs. A fractional factorial design with too few experimental runs

to allow estimation of all of the main effects is called a **supersaturated design (SSD)**. Specifically, these designs have n runs but the number of factors is $k > n - 1$ (Booth and Cox, 1962). If the number of factors of interest is very large and the experimenter wishes to identify which few may be the most important, then an SSD can be used for factor screening. Alternatively, if the experimental runs are very expensive to perform, it may not be practical to use a larger design (even a small fractional factorial design), and an SSD may be a more economical alternative.

For a design with k factors, the main effects only model can be written as

$$y = \beta_0 + \sum_{j=1}^{k} \beta_j x_j + \epsilon, \tag{1.3}$$

where y is the response variable, β_0, ..., β_k are unknown parameters, x_1, ..., x_k are the coded levels of the k factors, and ϵ is an error term that is normally distributed with mean $E(\epsilon) = 0$ and variance $Var(\epsilon) = \sigma^2$. The main effects only model can also be written in matrix notation as

$$y = \mathbf{X}\beta + \epsilon, \tag{1.4}$$

where y and ϵ are $n \times 1$ column vectors, β is a $(k+1) \times 1$ column vector, and $\mathbf{X}$ is an $n \times (k+1)$ matrix with a row for each experimental run, a column of 1's for the intercept and a column of coded levels for each factor. The matrix $\mathbf{X}$ is called the **model matrix**. The matrix that specifies the levels of the factors without any additional columns (such as the intercept) is called the **design matrix**. The ordinary least squares estimates for the coefficients, which minimize the sum of squared vertical distances between the observed and predicted responses (i.e. sum of squared residuals) are

$$\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'y. \tag{1.5}$$

By definition, the rank of $\mathbf{X}$ must be less than or equal to the minimum of n and $k+1$. Since the number of factors $k \geq n$, the rank of $\mathbf{X}$ can be at most n. The rank of $\mathbf{X}'$ is also then at most n, and therefore the rank of $\mathbf{X}'\mathbf{X}$ is at most n. But $\mathbf{X}'\mathbf{X}$ is a $(k+1) \times (k+1)$ matrix, and since $k + 1 > n$, it is not full rank. This means that $\mathbf{X}'\mathbf{X}$ is singular, and there will be no unique least squares estimate for the β coefficients. For this reason, it is necessary to pursue other methods of constructing and analyzing SSDs.

Chapter 2

Literature Review

2.1 Introduction and Background

In 1959, Satterthwaite published an article discussing random balance experimentation, where a random sampling process is used to determine which runs from a model matrix $\mathbf{X}$ are performed. For experiments with a large number of factors or a limited budget, random balance experiments were a relatively simple and efficient alternative to full factorial designs. In fact, Satterthwaite suggested that it may be desirable for a design to have more factors than runs (although this was not a requirement for a random balance experiment). A few years later, Booth and Cox (1962) formally introduced the idea of SSDs as factorial designs with n observations where the number of factors is greater than $n-1$. In other words, an SSD is defined as a design having at least as many factors as there are experimental runs, which means there are not enough degrees of freedom to estimate all of the main effects.

Booth and Cox (1962) proposed the SSD as a screening solution for experiments with a large number of potentially relevant factors, but where only a small proportion are thought likely to be significant. Today we call this effect sparsity, the assumption that only a few main effects and low-order interactions are important. Especially over the past ten to fifteen years, as processes have grown more complicated and experimentation has become more expensive, there has been growing interest in SSDs and their applications in factor screening. In fact, high dimensional data sets containing far more factors than observations are becoming very common today in areas such as genetics, biomedical imaging and finance (Fan and Lv, 2008).

SSD factors can have two or more levels, and it is important to note that work has been published about multi-level SSDs and related design criteria and construction methods (including Yamada and Lin (1999) on three-level designs, and Lu et al. (2003) on general

multi-level designs). Throughout the discussion that follows, however, we will remain focused on two-level designs with coded levels -1 and $+1$. When possible, we will require that for each factor, half the runs occur at the low level -1 and the other half occur at the high level $+1$. This balance is a desirable attribute of the full and fractional factorial designs that were discussed previously, although it is not always possible for SSDs. For consistency, we will continue to refer to the model matrix as $\mathbf{X}$.

2.2 Design Construction Methods

Holcomb and Carlyle (2002) classified methods of SSD construction into two categories: those that employ search techniques to develop designs, and those that are based on the structure of existing orthogonal designs.

2.2.1 Search Techniques

Suppose an experimental design in n runs and k factors is desired. One possible search technique would be to begin with the set of all possible columns of n runs where half the entries are $+1$ and half the entries are -1. Then any subset of k columns (plus an additional column of 1's for the intercept) could form a model matrix $\mathbf{X}$ (Holcomb and Carlyle, 2002). Next, the set of all possible design matrices could be compared in terms of some ranking criteria, and the most optimal design selected. However, depending on the number of factors and runs desired, the number of potential design matrices could be prohibitively large. Therefore, this method of construction and design selection is not ideal in most cases.

Consider the small SSD given in Table 2.1 that was suggested by Gilmour (2006).

Table 2.1: A small supersaturated design in $k = 10$ factors with $n = 6$ runs

Factors									
1	2	3	4	5	6	7	8	9	10
+	+	−	+	+	+	−	−	−	+
+	+	+	−	−	−	+	−	+	+
−	−	+	−	+	+	−	+	+	+
−	+	−	+	+	−	+	+	+	−
+	−	+	+	−	+	+	+	−	−
−	−	−	−	−	−	−	−	−	−

The corresponding $\mathbf{X}'\mathbf{X}$ matrix for this design is

$$\mathbf{X}'\mathbf{X} = \begin{bmatrix} 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 6 & 2 & 2 & 2 & -2 & 2 & 2 & -2 & -2 & 2 \\ 0 & 2 & 6 & -2 & 2 & 2 & -2 & 2 & -2 & 2 & 2 \\ 0 & 2 & -2 & 6 & -2 & -2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 2 & 2 & -2 & 6 & 2 & 2 & 2 & 2 & -2 & -2 \\ 0 & -2 & 2 & -2 & 2 & 6 & 2 & -2 & 2 & 2 & 2 \\ 0 & 2 & -2 & 2 & 2 & 2 & 6 & -2 & 2 & -2 & 2 \\ 0 & 2 & 2 & 2 & 2 & -2 & -2 & 6 & 2 & 2 & -2 \\ 0 & -2 & -2 & 2 & 2 & 2 & 2 & 2 & 6 & 2 & -2 \\ 0 & -2 & 2 & 2 & -2 & 2 & -2 & 2 & 2 & 6 & 2 \\ 0 & 2 & 2 & 2 & -2 & 2 & 2 & -2 & -2 & 2 & 6 \end{bmatrix}. \tag{2.1}$$

From the previous chapter, we know that because $\mathbf{X}'\mathbf{X}$ is not full rank, it is therefore singular and the inverse does not exist. Gilmour (2006) suggests the following:

> ...if it were possible to reduce the off-diagonal elements in absolute value, the matrix could be made nonsingular. Best of all would be if the off-diagonal elements were all zero, in which case all main effects would be estimated independently. (p.171)

One of the underlying problems with SSDs is that the design columns are not pairwise orthogonal. (So it is not true for every pair of columns, c_i and c_j, that $c_i'c_j = 0$.) This means there is some inherent correlation between the design columns. If the columns were pairwise orthogonal, then the off-diagonal elements that Gilmour (2006) refers to would all be zero.

Booth and Cox (1962) designed a computer algorithm and ran it on a University of London Mercury computer to calculate cross-products between pairs of trial vectors and iterate through various other trial vectors to look for ones that would improve the orthogonality of the model matrix columns. They addressed the concern about nonorthogonality in two ways. First, they required a minimal value for

$$\max_{i \neq j} |c_i'c_j|, \tag{2.2}$$

where c_i and c_j are the ith and jth columns of the model matrix $\mathbf{X}$. If two different SSDs had the same value for (2.2), then the one with the fewest pairs of columns attaining (2.2) would be preferred. Second, they suggested comparing designs using the expected value of the square of $s = c_i'c_j$, which can be calculated by taking the sum of squares of all the off-diagonal terms of $\mathbf{X}'\mathbf{X}$, and dividing by the number of off-diagonal terms. The formula

for this now popular criterion is given by

$$E(s^2) = \frac{\sum_{i<j} s_{ij}^2}{\binom{k}{2}} = \frac{2}{k(k-1)} \sum_{i<j} s_{ij}^2, \tag{2.3}$$

where k is the number of factors. Note that for designs where the mean of the off-diagonal elements of $\mathbf{X}'\mathbf{X}$ is $E(s) = 0$, $E(s^2)$ is simply the variance of s. Booth and Cox (1962) used their computer algorithm to iterate through various combinations of candidate design columns, evaluating the resulting designs in terms of these nonorthogonality criteria.

Li and Wu (1997) later suggested a design construction algorithm called "columnwise-pairwise exchange" that was based on D-optimality criteria (which seeks to maximize the determinant of the $\mathbf{X}'\mathbf{X}$ matrix) and has applications both for SSDs and also for designs that are not supersaturated. However, Holcomb and Carlyle (2002) noted that the designs developed by Booth and Cox (1962) and by Li and Wu (1997) generally did not allow for as many factors as some other designs with the same number of runs.

Lin (1995) took a slightly different approach to column-swapping search methods. For his algorithm, one first has to select the number of desired runs, n, and also a maximum acceptable value, r, for the degree of nonorthogonality between pairs of columns in the model matrix. For any pair of columns, c_i and c_j, the r_{ij} can be calculated by

$$r_{ij} = \frac{c_i' c_j}{n}. \tag{2.4}$$

Note that if two columns are orthogonal, then their $r_{ij} = 0$. Because we desire the columns to be as pairwise orthogonal as possible, we desire a maximum r value that is as close to zero as possible. Once the values for n and r have been selected, the set of all possible $n \times 1$ design columns is generated, with the requirement that half of the n runs are $+1$ and half are -1. In random order, candidate columns enter the design matrix. Each time, the r_{ij} values are computed between the new column and each of the existing design matrix columns. If the maximum r_{ij} is less than r, the new column stays in the design matrix. If the candidate column satisfies the r value for all design columns except one, the candidate column is saved in a queue for later assessment. If the r requirement is not satisfied for two or more design columns, the candidate column is discarded altogether. The algorithm iterates until the set of all possible design columns has been exhausted, with each candidate column landing in one of three places: the design matrix, the queue of saved columns, or the discard pile.

Next, the queue of saved columns is reviewed against the current design matrix. Whenever two of the queue columns meet the r requirement for all but the same retained design column,

that retained column will be replaced by the two columns from the queue. In this way, the algorithm continues to grow the design matrix until no other candidate columns will satisfy the r requirement for entry into the matrix. Note that to remove any selection bias, this algorithm can be rerun multiple times with the candidate columns entering the design matrix in different random orders.

The final step in Lin's (1995) search method is to order the columns so that those that are most nearly orthogonal to all the others appear first, which is determined by trying to minimize the average of r_{ij}^2, called the mean squared correlation, which is calculated by

$$\rho^2 = \frac{\sum r_{ij}^2}{\binom{k}{2}}. \tag{2.5}$$

Note that this criterion is equivalent to the widely accepted $E(s^2)$ criterion created by Booth and Cox (1962). This last step of column ordering is important so that if an experimenter needs n runs but the number of relevant factors, k, is less than the number generated by the algorithm for that run size, then the experimenter can simply select the first k columns (plus a column of 1's for the intercept) to form the size $n \times (k+1)$ model matrix $\mathbf{X}$. For example, p.216 of Lin's (1995) article gives a design with $n = 12$ runs and $k = 66$ factors. If an experiment were to require 12 runs but only 20 factors, then only the first 20 columns of Lin's (1995) design matrix would be used.

2.2.2 Construction from Existing Orthogonal Design Structure

Lin (1993) proposed a method for creating SSDs based on half fractions of Hadamard matrices. A Hadamard matrix of order N is an $N \times N$ orthogonal square matrix where every element is either -1 or $+1$ (Wu and Hamada, 2009). Lin's construction method begins with a Hadamard matrix of order N, where $N \leq 60$. Using the levels of a "branching column" (any one of the factor columns where half the entries are -1 and half are $+1$) to group the rows, the Hadamard matrix can be divided into two halves. One of the groups is selected, and the branching column is discarded, leaving a supersaturated design matrix with $N-2$ factors and $N/2$ runs. Lin suggests using the group where the branching column elements are $+1$.

Specifically, Lin (1993) suggested the use of Plackett-Burman designs, which are two-level fractional factorial designs developed by Plackett and Burman in 1946 for studying $k = n-1$ variables with n experimental runs (Plackett and Burman, 1946). All Plackett-Burman design matrices are $n \times n$ Hadamard matrices, which makes them ideal for use in this half-fraction Hadamard matrix construction method.

Lin (1993) showed an SSD construction example that uses the model matrix for the $n = 12$ Plackett-Burman design (an order $N = 12$ Hadamard matrix) shown in Table 2.2. The run number is shown for reference (although it is not part of the model matrix $\mathbf{X}$). Column I represents the intercept column of 1's, and the remaining columns show the factor levels of the $k = 11$ factors. Column I and the 11 factor columns form the model matrix $\mathbf{X}$.

Table 2.2: The Plackett-Burman model matrix with $k = 11$ factors and $n = 12$ runs (an order $N = 12$ Hadamard matrix)

	Factors											
Run	I	1	2	3	4	5	6	7	8	9	10	11
1	+	+	+	−	+	+	+	−	−	−	+	−
2	+	+	−	+	+	+	−	−	−	+	−	+
3	+	−	+	+	+	−	−	−	+	−	+	+
4	+	+	+	+	−	−	−	+	−	+	+	−
5	+	+	+	−	−	−	+	−	+	+	−	+
6	+	+	−	−	−	+	−	+	+	−	+	+
7	+	−	−	−	+	−	+	+	−	+	+	+
8	+	−	−	+	−	+	+	−	+	+	+	−
9	+	−	+	−	+	+	−	+	+	+	−	−
10	+	+	−	+	+	−	+	+	+	−	−	−
11	+	−	+	+	−	+	+	+	−	−	−	+
12	+	−	−	−	−	−	−	−	−	−	−	−

We divide the runs into two groups using factor 11 as the branching column. Runs 1, 4, 8, 9, 10 and 12 form the first group (factor 11 is -1 for all of these runs), and runs 2, 3, 5, 6, 7 and 11 form the second group (factor 11 is $+1$ for all of these runs). Since we want the group where the branching column was $+1$, we will use the second group of runs as an SSD with $n = N/2 = 6$ and $k = N - 2 = 10$. Table 2.3 shows the SSD that would result.

Wu (1993) also proposed a method for SSD construction using Hadamard matrices. We can create an interaction column c_{ij} from any two design columns c_i and c_j by multiplying them term by term. If an interaction column is not fully aliased with any existing column of the design matrix, then the column can be added to the design to study an additional factor. Wu found that for some small Plackett-Burman designs, the number of applicable interaction columns can be very large, which enables construction of an SSD where the number of factors, k, is much larger than the number of runs, n. Wu showed that an $N \times N$ Hadamard matrix can be used to construct a design for up to 2^{N-1} factors with just N runs.

Table 2.3: A supersaturated design with $k = 10$ factors and $n = 6$ runs (formed from a half-fraction of an $N = 12$ Hadamard matrix)

Factors									
1	2	3	4	5	6	7	8	9	10
+	−	+	+	+	−	−	−	+	−
−	+	+	+	−	−	−	+	−	+
+	+	−	−	−	+	−	+	+	−
+	−	−	−	+	−	+	+	−	+
−	−	−	+	−	+	+	−	+	+
−	+	+	−	+	+	+	−	−	−

2.3 Ranking Criteria

Several criteria exist for comparing SSDs. As mentioned in section 2.2.1, one challenge with SSDs is that their columns are not pairwise orthogonal. Therefore, some criteria measure the degree of orthogonality of the design.

As discussed previously, Booth and Cox (1962) ranked their designs by assessing the relative orthogonality of the design matrices. They considered the value of

$$\max_{i \neq j} |c_i' c_j|, \tag{2.6}$$

where c_i and c_j are the ith and jth columns of the model matrix $\mathbf{X}$. If two designs had the same value for (2.6), the one with the fewest pairs of columns attaining (2.6) is preferred.

Booth and Cox (1962) also suggested comparing designs using the expected value of the square of $s = c_i' c_j$, which can be calculated by taking the sum of squares of all the off-diagonal terms of $\mathbf{X}'\mathbf{X}$, and dividing by the number of off-diagonal terms, as given by

$$E(s^2) = \frac{\sum_{i<j} s_{ij}^2}{\binom{k}{2}} = \frac{2}{k(k-1)} \sum_{i<j} s_{ij}^2, \tag{2.7}$$

where k is the number of factors. This criterion is one of the most popular ranking criteria for SSDs. Note that if the design is orthogonal, then all s_{ij} off-diagonal elements of $\mathbf{X}'\mathbf{X}$ are equal to zero, which means $E(s^2) = 0$. Also note that for designs where the mean of the

off-diagonal elements of $\mathbf{X}'\mathbf{X}$ is $E(s) = 0$, $E(s^2)$ is simply the variance of s.

Nguyen (1996) later showed that, in general, the optimal value for $E(s^2)$ is

$$E(s^2) = \frac{n^2(k-n+1)}{(k-1)(n-1)}, \tag{2.8}$$

where n is the number of experimental runs and k is the number of factors. A design that achieves this optimal value is called an $E(s^2)$-optimal design. The ratio of $E(s^2)$ to optimal $E(s^2)$ can be computed and used as another design ranking criteria (Holcomb and Carlyle, 2002).

Booth and Cox (1962) also showed that for a design with k effects, the average variance for any estimated effect will be approximately

$$\frac{\sigma^2}{n}\left\{1 + \frac{(k-1)E(s^2)}{n^2}\right\}. \tag{2.9}$$

An $E(s^2)$-optimal design would also minimize this quantity (Gilmour, 2006).

For cases where all factors are considered to be roughly equally important (instead of some factors being presumed to be more important than others), Holcomb and Carlyle (2002) also suggested an eigenvalue ratio criterion, κ_r, which is the ratio of the largest to the smallest non-zero eigenvalues of the $\mathbf{X}'\mathbf{X}$ matrix. When $\mathbf{X}'\mathbf{X}$ is rank n, the model matrix will have n non-zero eigenvalues and $k-n$ zero eigenvalues (Holcomb and Carlyle, 2002). A small eigenvalue or eigenvalues would indicate that several columns are nearly linearly dependent, in which case the design is said to have poor κ_r-optimality. Because κ_r is the ratio of the largest non-zero eigenvalue to the smallest non-zero eigenvalue, an unusually small eigenvalue (compared to the other eigenvalues) would cause κ_r to be very large. We desire the magnitude of the κ_r ratio to be as close to 1 as possible.

Wu (1993) developed A_f and D_f optimality criteria to rank SSDs. The D_f criterion is calculated by

$$D_f = \frac{\sum_i |\frac{1}{n}X_i'X_i|^{1/f}}{\binom{k}{f}}, \tag{2.10}$$

where n is the number of runs, k is the number of factors, f is the number of active factors

(i.e. the number of factors that truly have significant influence on the response), and $\{X_i\}$ is the set of all $n \times f$ submatrices of $\mathbf{X}$. The similar A_f criterion is calculated by

$$A_f = \frac{\sum_i \frac{1}{f} tr(\frac{1}{n} X_i' X_i)^{-1}}{\binom{k}{f}}. \tag{2.11}$$

Wu (1993) notes that $A_f > 1$ in general, and is equal to 1 when the design is orthogonal. Therefore, $A_f - 1$ can be used as another way to measure the nonorthogonality of a design. Nguyen (1996) later showed that the D_f and A_f criteria are approximately equivalent and typically lead to a consistent design ranking.

Jones et al. (2008) discussed ranking designs in terms of a Bayesian D-optimality criterion. For an SSD, factors would be categorized into two groups – a "primary" group that includes the intercept and k_1 factors that are presumed to be active, and a "potential" group of the remaining $k_2 = k - k_1$ factors that may or may not be active. A prior mean of zero and variance τ^2 are assumed for the k_2 potential factors. The Bayesian D-optimality criterion is

$$D_{Bayes} = \left| \mathbf{X}'\mathbf{X} + \frac{\mathbf{K}}{\tau^2} \right|, \tag{2.12}$$

where

$$\mathbf{K} = \begin{bmatrix} 0_{k_1 \times k_1} & 0_{k_1 \times k_2} \\ 0_{k_2 \times k_1} & I_{k_2 \times k_2} \end{bmatrix}. \tag{2.13}$$

A Bayesian D-optimal design with n runs and k factors would be the design of that size that maximizes the D_{Bayes} criterion (Jones et al., 2008).

2.4 Analysis Methods

As discussed in the previous chapter, there are no unique ordinary least squares estimates for the β parameters in the main effects only model for an SSD. Also, because there are fewer degrees of freedom available than there are effects to estimate, there is no way to estimate all of the main effects simultaneously. When analyzing an SSD, one has to rely heavily on the principal of effect sparsity and assume that only a few factors are significant. The goal of data analysis for any SSD is to screen for the most important main effects.

Satterthwaite (1959) suggested that the significance of each main effect could be reviewed separately by a single factor simple linear regression. In other words, the effect of factor 1 could be estimated by fitting the model

$$y = \beta_0 + \beta_1 x_1 + \epsilon, \tag{2.14}$$

and similarly for factors 2 through k. This method, however, does not take into account any redundancy of effects, or joint correlations between effects (i.e. effects that are important in the presence of others, but not on their own).

Holcomb et al. (2003) extended the simple linear regression idea into a "many models" method. First, all two-factor main effects models would be created. An example of such a model (using factors 1 and 2) is

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon. \tag{2.15}$$

These two-factor models would be ranked in terms of their error sums of squares, and the best models would be selected. (The process description is not specific about how many "best" models should be retained, but presumably this would vary based on the range of error sums of squares.) Once the best models are selected, we would order the individual factors in terms of how frequently they appear in the retained best models. Any factors that appear in at least 75% of the best models would be retained and used to generate three-factor main effects models. Those three-factor models would then be ordered, the best ones selected, and the process would continue accordingly until either the desired number of factors have been estimated, or until there are no other significant factors to add to the "best" model.

Motivated by the "many models" approach, Marley and Woods (2010) advocated a model averaging procedure, where coefficient estimates would be obtained by calculating estimates for a large set of models, and then computing a weighted average of all the estimates for each coefficient. The weights would be assigned according to the plausibility of each model.

Holcomb et al. (2003) also discussed using stepwise model selection similar to that used in normal multiple linear regression. The process would begin by fitting the full series of single-factor simple linear regression models. The factor that results in the model with the lowest error sum of squares and has a significant partial F-test would be retained. In a similar fashion, all possible second factors would be added to and tested in the model one at a time, and the one that results in the lowest error sum of squares would be retained. Holcomb et al. (2003) suggested that after the model contains three or more factors by this

process, backwards elimination would be used to confirm the continued significance of each model term. If any previously-added term was later determined to be insignificant in the model (based on a partial F-test), the factor in question would be dropped from the model. Factors would continue to be added and dropped from the model in alternating fashion until no additional factors meet the criteria for entry into the model, and no retained factors fail to meet the criteria for retention in the model.

For stepwise processes, one can use a specific p-value (for example, $\alpha = 0.01$ or $\alpha = 0.05$) to determine whether factors can enter or exit the model. Alternatively, the Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC) can also be used, which are relative measures of the goodness of fit of a model. The AIC is defined as

$$AIC = 2p - 2\ln(L) + \frac{2p(p+1)}{n-p-1}, \tag{2.16}$$

where p is the number of model parameters ($p = k + 1$), n is the number of observations, and L is the maximized value of the likelihood function for the estimated model. The BIC is defined as

$$BIC = -2\ln(L) + p\ln(n), \tag{2.17}$$

where p is the number of model parameters, n is the number of observations, and L is the maximized value of the likelihood function for the estimated model. In both cases, the "best" model is the one that minimizes the AIC or BIC value, so in a stepwise selection process, factors could enter or exit the model according to what would reduce AIC or BIC the most.

Abraham et al. (1999) suggested an all-subsets variable selection approach, which would evaluate every possible subset of factors in terms of the R^2 value achieved by regressing the model against the response. Obviously this could become cumbersome as the number of factors increases. Compared to stepwise regression, this method would certainly evaluate more candidate models; however, Abraham et al. (1999) determined that even an all-subsets method can be problematic because the process is cumbersome and also does not provide a clear way of deciding how many factors are active.

Candes and Tao (2007) introduced the Dantzig selector, a now popular "shrinkage" type of selection method for SSDs that is the solution to the l_1-regularization problem

$$\underset{\hat{\beta} \in R^k}{min} ||\hat{\beta}||_{l_1}, \tag{2.18}$$

subject to the constraint that

$$||\mathbf{X}'(y - \mathbf{X}\hat{\beta})||_{l_\infty} \leq \delta, \tag{2.19}$$

where δ is a tuning parameter, and for a vector v, $||v||_{l_1} = \sum |v_i|$ and $||v||_{l_\infty} = \max |v_i|$ (Phoa et al., 2009). Once the solution $\hat{\beta}$ has been identified, a set of factors $\hat{I} = \{i : |\hat{\beta}_i > \gamma|\}$ would be chosen for some $\gamma \geq 0$, and a second estimator could be created by

$$\hat{\beta}_{\hat{I}} = (\mathbf{X}'_{\hat{I}}\mathbf{X}_{\hat{I}})^{-1}\mathbf{X}'_{\hat{I}}y, \tag{2.20}$$

setting the other coefficients to zero (Phoa et al., 2009). Candes and Tao (2007) called this second estimator the Gauss-Dantzig selector.

Lin (1995) suggested that ridge regression techniques can be used to determine the most influential factors; however, for cases where the number of factors, k, is much larger than the number of experimental runs, n, he found that ridge regression did not work very well. (He did not, however, specify an ideal maximum for the k to n ratio.)

Koh et al. (2011) used a permutation test method for SSD variable selection. This method involves rearranging the responses of a data set and computing the test statistic for all possible permutations (or a large number of permutations, if the number of responses, n, is very large).

Other SSD analysis techniques exist other than the ones that have been reviewed here, including additional Bayesian variable selection techniques suggested by Chipman et al. (1997).

Unfortunately, existing methods for analysis of SSDs are known to typically have very large type I or type II error rates (or a moderate rate of both type I and type II). Since SSDs are used for factor screening, in most cases, a type I error would be preferred over a type II error, since it would be better to falsely identify an inactive factor as active than to miss a truly active factor altogether. Holcomb et al. (2003) suggested that the type I error rate for an SSD could be as high as 50% just so that the type II error rate will be moderate. Due to the high likelihood of type I errors, it is essential that any SSD be followed by an additional small factorial or fractional factorial experiment to validate the analysis results, and to also incorporate low-order interaction terms.

In theory, SSDs are useful in situations where budgets are limited and/or there are lots of potential factors to investigate. While there are several clever ways to construct SSDs,

the associated analysis methods are error-prone and can produce unreliable results. In the remainder of this paper, we will examine an additional analysis method, Sure Independence Screening (SIS) to examine its usefulness for analyzing SSDs.

Chapter 3

The Sure Independence Screening Approach

3.1 Penalized Least Squares and the Smoothly Clipped Absolute Deviation (SCAD) Penalty

For the linear regression model given by $y = \mathbf{X}\beta + \epsilon$ (with error ϵ distributed as $N(0, \sigma^2)$), the least squares estimate for the β vector of coefficients is obtained by minimizing the value of the sum of squared residuals. Since the response y is estimated by $\hat{y}$, and the expected value of the error term ϵ is zero, the sum of squared residuals is $||y - \mathbf{X}\beta||^2$ (Fan and Li, 2001).

Fan and Li (2001) introduced a form of penalized least squares that can be written as

$$Q(\beta) = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \mathbf{X}_i'\beta)^2 + \sum_{j=0}^{k} p_{\lambda_j}(|\beta_j|), \tag{3.1}$$

where $p_{\lambda_j}(\cdot)$ is a penalty function, and λ_j is a tuning parameter. The tuning parameter can be chosen by a method such as cross-validation (CV) or generalized cross-validation (GCV) (Li and Lin, 2002), or even by minimizing AIC or BIC. Fan and Li (2001) outlined several desirable properties that estimators produced by good penalty functions should possess: unbiasedness, sparsity (where small estimators are set to zero to avoid unnecessary model complexity), and continuity (having the estimator be continuous over the data set to avoid instability in model prediction).

Fan and Li (2001) advocated the Smoothly Clipped Absolute Deviation (SCAD) penalty, which has first-order derivative

$$p'_\lambda(\beta) = \lambda \left\{ I(\beta \leq \lambda) + \frac{(a\lambda - \beta)_+}{(a-1)\lambda} I(\beta > \lambda) \right\}, \tag{3.2}$$

where $\beta > 0$, $a > 2$, $p_\lambda(0) = 0$, and $I(\cdot)$ is an indicator function. They recommended the value $a = 3.7$ based on a Bayesian risk analysis they performed to identify the best value for a, so this value is typically used in the SCAD penalty function.

Li and Lin (2002) noted that the SCAD penalty function is singular at the origin and the second derivative may not exist at some points. Therefore, they used an iterative ridge regression technique and a local quadratic approximation for the penalty function to determine the final solution for the SCAD-penalized least squares. They approximated the SCAD penalty function by the quadratic function

$$[p_\lambda(|\beta_j|)]' = p'_\lambda(|\beta_j|)sign(\beta_j) \approx \left(\frac{p'_\lambda(|\beta_j^{(0)}|)}{|\beta_j^{(0)}|} \right) \beta_j, \tag{3.3}$$

where $\beta^{(0)}$ is an initial value of the coefficient estimates. Using the local quadratic approximation function and initial value $\beta^{(0)}$, the final solution for the SCAD-penalized least squares can be found by using iterative ridge regression as given by

$$\beta^{(1)} = \left\{ \mathbf{X}'\mathbf{X} + n \sum_\lambda (\beta^{(0)}) \right\}^{-1} \mathbf{X}'y, \tag{3.4}$$

where

$$\sum_\lambda (\beta^{(0)}) = diag \left\{ \frac{p'_\lambda(|\beta_1^{(0)}|)}{|\beta_1^{(0)}|}, ..., \frac{p'_\lambda(|\beta_k^{(0)}|)}{|\beta_k^{(0)}|} \right\}. \tag{3.5}$$

The only thing remaining is to find the initial value for $\beta^{(0)}$ that is close to the true value of β. If the model matrix was full rank, we could simply use the least squares estimates of the coefficients. However, because the model matrix for an SSD cannot be full rank, most typically a stepwise variable selection method is used to find the initial value. A high significance value is recommended to ensure that all of the active factors make it into the

model (i.e. to avoid type II errors in the initial $\beta^{(0)}$ selection). Li and Lin (2002) used significance level 0.1 in their stepwise variable selection process for choosing an initial value for their SCAD simulation, which we will discuss further in Chapter 4.

3.2 Sure Independence Screening (SIS)

Fan and Lv (2008) proposed a screening method called **sure independence screening (SIS)**, where the desired outcome is that all of the active factors survive the screening process. Their method uses componentwise regression to identify important factors one at a time, based on their individual correlations with the response variable. Specifically, the process begins by standardizing each factor column of the $n \times p$ model matrix $\mathbf{X}$ (where $p = k + 1$ is the number of β parameters in the model) to have a mean of zero and standard deviation of one. Then a $p \times 1$ vector ω is formed by componentwise regression as given by

$$\omega = \mathbf{X}'y, \tag{3.6}$$

where $\mathbf{X}$ is the standardized model matrix and y is the $n \times 1$ vector of responses. Fan and Lv (2008) pointed out that ω is really then a vector of marginal correlations of the main effects against the response. The k factor components of ω can be sorted by absolute magnitude, from highest to lowest, and some $\gamma \in (0, 1)$ percentage selected to form a set of factors

$$M_\gamma = \{1 \leq i \leq k : |\omega_i| \text{ is among the first } [\gamma n] \text{ largest}\}, \tag{3.7}$$

where $[\gamma n]$ denotes the integer portion of γn, if it is not already an integer. The level of γ can be chosen so that it reduces the model size to less than n factors. For example, we can select the maximum estimable number of factors, $n - 1$, by choosing $\gamma = (n - 1)/n$.

Note that after performing SIS to reduce the set of all possible factors to one of size less than n, SCAD can be applied to estimate the coefficients and screen out those that are not significant in the presence of the others – i.e. that have coefficient estimates near zero. This may further reduce the number of significant factors, and will provide the estimates that are necessary for writing out the fitted model. This combined technique is called SIS-SCAD.

Fan and Lv (2008) acknowledged three potential flaws with the sure independence screening (SIS) approach. First, effects that are not themselves important to the response, but are highly correlated with other effects that *are* important to the response, would be falsely identified as being important in the model. A second related issue is the potential for collinearity

between factors in the final model, since they are considered one by one during the screening process. Thirdly, a truly important factor that is jointly correlated with the response in the presence of some other factor, but that alone is uncorrelated with the response, will not make it into the model.

Essentially, these three issues imply that using SIS for factor screening will result in high type I and type II error rates. This is common with SSDs, as we have already discussed. Note that following SIS with SCAD allows the down-selected set of up to $n - 1$ variables to be analyzed together, which may eliminate some of the correlation and collinearity issues from the initial SIS component-wise regression. It does not, however, address the potential issue with type II errors (i.e. active factors being missed). Fan and Lv (2008) attempted to resolve this issue by introducing an iterative component to the original SIS technique.

3.3 Iterative Sure Independence Screening (ISIS)

Fan and Lv (2008) proposed the following iterative process to alleviate some of the aforementioned challenges with the SIS variable selection approach. First, a subset of k_1 factors would be identified using SIS followed by a model fitting process such as SCAD. We can refer to this initial subset of factors as $A_1 = \{X_{i_1}, ..., X_{i_{k_1}}\}$. Note that this initial round of SIS-SCAD will produce an n-vector of residuals. SIS-SCAD is then applied to the remaining $k - k_1$ factors, using the residuals from the first regression as the response variable. This second round of SIS-SCAD will generate another subset of k_2 factors, $A_2 = \{X_{j_1}, ..., X_{j_{k_2}}\}$, and a new set of n residuals.

Note that by using the residuals as the new response variable, this helps to resolve the third challenge with the non-iterative SIS process. That is, with one-step SIS, an important factor that alone is uncorrelated with the response, but that is jointly correlated with the response in the presence of some other factor(s), would not be selected into the model. The factor in question is, however, likely to be correlated with the residuals from the first regression, if it is jointly correlated with some other factor that has been brought into the model, and it would be pulled into the second set of factors, A_2.

After the second subset of factors, A_2, has been found, the SIS-SCAD process is then applied to the remaining $k - k_1 - k_2$ variables, using the newest set of residuals as the response. This process would continue until l disjoint subsets are obtained, $A_1, ..., A_l$. In practice, a constraint would be placed on l such that the size of the union of sets $A = \cup_{i=1}^{l} A_i$ will be less than n. Once the final set of variables, A, has been identified, SCAD is applied one final time to estimate the significant coefficients and to discard those that are not significant in the presence of the others. This generates the final list of significant factors and the fitted model.

Chapter 4

Simulation Studies

4.1 Williams Rubber Data

4.1.1 Comparison of SCAD and Stepwise Variable Selection Procedures by Li and Lin (2002)

In 2002, Li and Lin performed a simulation study to compare the SCAD procedure against stepwise variable selection procedures. They used the design matrix constructed by Lin (1993), which is shown in Table 4.1. This design is a half-fraction of the 24 factor, 28 run Plackett-Burman design originally used in Williams's rubber data study (1968). Note that in the 24 factor half-fraction design, the levels of factors 13 and 16 are identical, and therefore factor 16 is omitted, with the remaining factors 17 to 24 being renumbered as 16 to 23. The resulting 23 factor, 14 run design is an $E(s^2)$-optimal SSD and is a popular design for testing analysis methods for SSDs.

Using this design, Li and Lin (2002) simulated 1000 data sets from each of the following three models:

$$y = 8X_1 + 5X_{12} + \epsilon \tag{4.1}$$

$$y = 10X_1 + 9X_2 + 2X_3 + \epsilon \tag{4.2}$$

$$y = -20X_1 + 12X_3 + 10X_5 + 5X_7 + 2X_{16} + \epsilon, \tag{4.3}$$

where the random error ϵ for each is distributed as $N(0, 1)$.

Table 4.1: Half-fraction of the Williams rubber data 28-run Plackett-Burman design, as suggested by Lin (1993), with duplicate factor 16 removed (and columns 17 - 24 renumbered)

Factors																						
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
+	+	+	−	−	−	+	+	+	+	+	−	+	−	−	+	−	−	+	−	−	−	+
+	−	−	−	−	−	+	+	+	−	−	−	+	+	+	−	+	−	−	+	+	−	−
+	+	−	+	+	−	−	−	−	+	−	+	+	+	+	+	−	−	−	−	+	+	−
+	+	−	+	−	+	−	−	−	+	+	−	+	−	+	−	+	+	+	−	−	−	−
−	−	+	+	+	+	−	+	+	−	−	−	+	−	+	+	−	−	+	−	+	+	+
−	−	+	+	+	+	+	−	+	+	+	−	−	+	+	+	+	+	+	+	+	−	−
−	−	−	−	+	−	−	+	−	+	−	+	+	+	−	+	+	+	+	+	−	−	+
−	+	+	−	−	+	−	+	−	+	−	−	−	−	−	−	−	+	−	+	+	+	−
−	−	−	−	−	+	+	−	−	−	+	+	−	−	+	+	+	−	−	−	−	+	+
+	+	+	+	−	+	+	+	−	−	−	+	−	+	+	+	−	+	−	+	−	−	+
−	+	−	+	+	−	−	+	+	−	+	−	−	+	−	−	+	+	−	−	−	+	+
+	−	−	−	+	+	+	−	+	+	+	+	+	−	−	−	−	+	−	+	+	+	+
+	+	+	+	+	−	+	−	+	−	−	+	−	−	−	−	+	−	+	+	−	+	−
−	−	+	−	−	−	−	−	−	−	+	+	−	+	−	−	−	−	+	−	+	−	−

Li and Lin (2002) performed a SCAD variable selection procedure, where the initial starting value was selected by stepwise variable selection with significance level $\alpha = 0.1$ for both entry into and removal from the model. In addition, three stepwise variable selection procedures (0.05 level of significance, best AIC and best BIC) were applied to the simulated data sets. All four methods were then compared in terms of the rate at which they identified the true models (presumably without including any extraneous factors), and also in terms of the size of the selected models. The results are shown in Table 4.2.

The results show that the SCAD procedure was much more effective at identifying the true model than the three stepwise procedures. The SCAD procedure picked the correct model over 70% of the time for all three models, whereas the stepwise procedures picked the right model less than 40% of the time. The mean and median for the SCAD procedures are close to the correct number of significant factors (2, 3 and 5), which demonstrates that SCAD did not include any inactive factors most of the time. The stepwise procedures, on the other hand, selected on average at least one extra factor that was not truly significant. Based on these results, Li and Lin (2002) concluded that SCAD is a viable variable selection procedure for use in the analysis of SSDs.

Table 4.2: Original results of Li and Lin (2002) simulated comparison of SCAD and stepwise variable selection methods

Model equation	Method	Rate of the true model being identified	Avg. size of fitted model	
			Median	Mean
(4.1)[a]	Stepwise (p = 0.05)	27.00%	3	3.7
	Stepwise (AIC)	2.20%	5	5.0
	Stepwise (BIC)	11.10%	4	4.2
	SCAD	82.70%	2	2.2
(4.2)[b]	Stepwise (p = 0.05)	32.30%	4	4.5
	Stepwise (AIC)	3.00%	6	5.74
	Stepwise (BIC)	13.50%	5	5.00
	SCAD	74.70%	3	3.34
(4.3)[c]	Stepwise (p = 0.05)	38.90%	6	6.17
	Stepwise (AIC)	7.00%	7	7.01
	Stepwise (BIC)	24.70%	6	6.43
	SCAD	71.90%	5	5.39

[a]Model (4.1): $y = 8X_1 + 5X_{12} + \epsilon$
[b]Model (4.2): $y = 10X_1 + 9X_2 + 2X_3 + \epsilon$
[c]Model (4.3): $y = -20X_1 + 12X_3 + 10X_5 + 5X_7 + 2X_{16} + \epsilon$

4.1.2 Comparison of SCAD, Stepwise and Permutation Variable Selection Procedures by Koh, Eskridge and Wang (2011)

Koh et al. (2011) performed a similar simulation study to that of Li and Lin (2002). They used the same design from Lin (1993) and the same model equations (4.1), (4.2) and (4.3), and compared three variable selection methods – permutation test, stepwise and SCAD – in terms of five criteria:

1. Identification rate (IR) - Percentage of simulations where a method identified exactly all the active factors, but no inactive factors
2. Power (P) - Average percentage of active factors that were correctly identified as active
3. Coverage (C) - Percentage of simulations where a method identified all the active factors and possibly a few other factors
4. Mean (M) - Average number of factors identified as being active
5. Variance (V) - Variance of the number of factors identified as being active

Unlike Li and Lin (2002), Koh et al. (2011) simulated data from the three models using several different error distributions, including standard normal $N(0,1)$, $t(3)$, beta$(2,2)$, exponential(1), uniform$(0,1)$, and standard normal $N(0,1)$ with outliers from $N(10,1)$. Their goal was to compare the usefulness of the chosen variable selection methods for predicting models with different error distributions. For purposes of this paper, we will focus on the results where the error was distributed as standard normal $N(0,1)$.

For each model, 1000 data sets were simulated and analyzed using the permutation test (0.01 and 0.05 levels of significance), stepwise selection method (0.01 and 0.05 levels of significance), and SCAD (using the same iterative ridge regression process from Li and Lin (2002)). Once again, the initial starting values for the SCAD procedure were found by stepwise selection with significance level $\alpha = 0.1$. The simulation results from these five variable selection methods are shown in Table 4.3.

Table 4.3: Original results of Koh, Eskridge and Wang (2011) simulated comparison of SCAD, stepwise and permutation test variable selection methods with $N(0,1)$ error distribution

Model	Method	IR	P	C	M	V
	Permutation test ($\alpha = 0.01$)	85	100	100	2.2	0.29
	Stepwise ($\alpha = 0.01$)	80	100	100	2.2	0.28
$(4.1)^a$	Permutation test ($\alpha = 0.05$)	40	100	100	3.0	1.09
	Stepwise ($\alpha = 0.05$)	26	100	100	3.9	3.48
	SCAD	88	100	100	2.2	0.23
	Permutation test ($\alpha = 0.01$)	73	95	85	3.1	0.53
	Stepwise ($\alpha = 0.01$)	79	100	100	3.3	0.29
$(4.2)^b$	Permutation test ($\alpha = 0.05$)	41	99	96	3.9	1.18
	Stepwise ($\alpha = 0.05$)	25	100	100	4.9	3.30
	SCAD	82	89	86	2.7	0.92
	Permutation test ($\alpha = 0.01$)	77	94	79	4.8	0.63
	Stepwise ($\alpha = 0.01$)	82	100	100	5.2	0.27
$(4.3)^c$	Permutation test ($\alpha = 0.05$)	57	98	92	5.3	0.78
	Stepwise ($\alpha = 0.05$)	33	100	100	6.5	2.74
	SCAD	62	72	62	3.6	4.55

[a]Model (4.1): $y = 8X_1 + 5X_{12} + \epsilon$
[b]Model (4.2): $y = 10X_1 + 9X_2 + 2X_3 + \epsilon$
[c]Model (4.3): $y = -20X_1 + 12X_3 + 10X_5 + 5X_7 + 2X_{16} + \epsilon$

These results show that the permutation and stepwise procedures (with $\alpha = 0.01$) and SCAD had 70% or higher identification rates, except for the SCAD application in model 3. This model has a combination of different sized coefficients, and it is possible that SCAD failed to find the small effects some of the time. Indeed, the average model size selected by SCAD

for model 3 was only 3.6 when there are actually five active factors. This is slightly different than the results of Li and Lin (2002), who reported an identification rate for model 3 of nearly 72%, with median and mean average fitted model sizes of 5 and 5.39, respectively. The identification rate for Koh et al. (2011) was only 62%.

The permutation and stepwise procedures with $\alpha = 0.05$ had lower identification rates, but high coverage and power, which indicates that these methods (with $\alpha = 0.05$) are prone to over-selection. This is likely consistent with the simulations of Li and Lin (2002), although they only reported on the identification rate, and not coverage and power.

4.1.3 Comparison of SIS and ISIS Variable Selection Procedures

We conducted a simulation study to examine the usefulness of SIS and ISIS variable selection procedures on the same 23 factor, 14 run half-fraction Plackett-Burman design suggested by Lin (1993) from the Williams rubber data study, and model equations (4.1), (4.2) and (4.3). The statistical package R 2.14.1 was used to generate 1000 data sets from these three models, with standard normal $N(0, 1)$ error distribution. All of the R code used in our data generation and analysis is shown in the Appendix.

To analyze the data, we used both SIS and ISIS variable selection procedures to select candidate factors, and followed both with SCAD parameter estimation approaches to down-select to the final set of variables. The R package 'SIS' was used for the SIS, ISIS, SIS-SCAD and ISIS-SCAD procedures and analysis (Fan et al., 2010).

For the SCAD tuning parameter λ, we used both AIC and BIC selection methods to see if and how it would affect the results. The prediction capabilities of the methods were examined with six criteria, similar to Koh et al. (2011), with one addition (type I error rate):

1. Identification rate (IR) - Percentage of simulations where a method identified exactly all the active factors, but no inactive factors
2. Coverage (C) - Percentage of simulations where a method identified all the active factors and possibly a few other factors
3. Power (P) - Average percentage of active factors that were correctly identified as active
4. Type I Error Rate (T1) - Average percentage of inactive factors that were incorrectly identified as active
5. Mean (M) - Average number of factors identified as being active

6. Variance (V) - Variance of the number of factors identified as being active

The results of our simulation study are shown in Table 4.4.

Table 4.4: Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods

Model	Method	AIC Tuning						BIC Tuning					
		IR	C	P	T1	M	V	IR	C	P	T1	M	V
(4.1)[a]	SIS	0	100	100	48	12.0	0.00	0	100	100	48	12.0	0.00
	SIS-SCAD	20	99	100	20	6.1	9.37	34	99	100	13	4.8	7.97
	ISIS	9	100	100	20	6.3	5.07	18	100	100	16	5.4	5.55
	ISIS-SCAD	9	100	100	20	6.1	4.76	18	100	100	16	5.3	5.19
(4.2)[b]	SIS	0	14	71	54	13.0	0.00	0	14	71	54	13.0	0.00
	SIS-SCAD	3	14	71	44	11.0	5.05	3	14	71	43	10.7	5.85
	ISIS	9	78	93	24	7.6	6.15	19	79	93	19	6.7	7.08
	ISIS-SCAD	9	77	92	23	7.3	5.89	19	78	93	19	6.5	6.76
(4.3)[c]	SIS	0	0	80	50	13.0	0.00	0	0	80	50	13.0	0.00
	SIS-SCAD	0	0	80	46	12.2	0.53	0	0	80	45	12.1	0.58
	ISIS	12	87	97	20	8.5	4.46	21	87	97	17	7.8	4.74
	ISIS-SCAD	12	86	97	19	8.3	4.28	22	86	97	16	7.7	4.47

[a]Model (4.1): $y = 8X_1 + 5X_{12} + \epsilon$
[b]Model (4.2): $y = 10X_1 + 9X_2 + 2X_3 + \epsilon$
[c]Model (4.3): $y = -20X_1 + 12X_3 + 10X_5 + 5X_7 + 2X_{16} + \epsilon$

One item of note is that the R 'SIS' package requires the specification of the maximum number of significant factors to return from the SIS and ISIS procedures. For model (4.1), we could not get the code to return any results when we selected $n - 1 = 13$ factors as the maximum; therefore, we chose to have the model find up to $n - 2 = 12$ factors. It was not clear from looking at the error message or within the 'SIS' package code why we received such an error, and this is suggested as an area of follow-up for future studies. For models (4.2) and (4.3), we were able to select up to $n - 1 = 13$ factors.

Also note that the single iteration SIS method will always return exactly the number of factors that we requested ($n - 1$ or $n - 2$ in this case, depending on the model in question). The SIS procedure performs the componentwise regression of the individual factors on the response, sorts the factors in terms of their correlation with the response, and then returns the number of factors we selected. Therefore, the mean number of factors returned by SIS will always be $n-1$ or $n-2$ (again, depending on the model), and the variance of the number of factors returned will always be zero. Models selected by SIS will therefore always suffer

from over-selection, unless the true number of active factors is the number we selected. Once SIS is followed by SCAD, the number of factors may be reduced.

For model (4.1), a very sparse model with one large and one moderate effect, the SIS-SCAD procedure had the highest identification rate – higher, in fact, than ISIS-SCAD. This is not what we expected and did not occur for the other two models. Since all four procedures have very high coverage and power, it appears that the iterative aspect of ISIS must have inadvertently pulled more inactive factors into the model than the original SIS process. This is confirmed by the larger mean number of factors returned by ISIS and ISIS-SCAD than by SIS-SCAD, although SIS-SCAD had higher variation in the number of factors selected. The average number of factors returned by SIS-SCAD and ISIS-SCAD were between 4.8 and 6.1, which means these procedures tend to overselect for very sparse models – indeed, the type I error rates are between 13% - 20%. However, the high coverage and power indicate that they nearly always include all the active factors (i.e. negligible type II error rate).

For models (4.2) and (4.3), which both have a combination of large and small effects, ISIS-SCAD had better identification rates than SIS-SCAD, although it was still only 22% or less. ISIS-SCAD had a good coverage rate of 77% - 86%, and high power also of 92% - 97%. Therefore, it appears that the majority of the time, ISIS-SCAD succeeds at finding the active factors, plus a few more. SIS-SCAD, on the other hand, had pretty good power (71% - 80%), but the coverage was low (14% for model (4.2) and 0% for model (4.3)). This suggests that there is some joint correlation between the factors against the response that ISIS was able to pick up, but the initial SIS iteration was not. This is one of the benefits of adding the iterative aspect of ISIS that we expected.

We were also interested in seeing if there was a difference between using an AIC or BIC selection method for the tuning parameter, λ. In fact, AIC and BIC both produced almost identical coverage and power results; however, the identification rate with the BIC-selected λ was slightly higher than with the AIC-selected λ, and the mean number of factors chosen with the BIC-selected λ was slightly lower than with the AIC-selected λ. The BIC selection method for λ was also recommended by Fan and Lv (2008), and so for future studies this tuning method is suggested.

Compared to the studies of stepwise-SCAD variable selection from Li and Lin (2002) and Koh et al. (2011), SIS-SCAD and ISIS-SCAD were less successful in identifying the exact models. The identification rates for the previous studies ranged upwards of 60% for stepwise-SCAD, whereas ours ranged from 0% - 34%.

Our power and coverage rates, however, were similar to those reported by Koh et al. (2011). For model 1, which had the greatest effect sparsity, coverage and power for all methods were nearly 100%, which is consistent with Koh et al. (2011) for the permutation test, stepwise selection and SCAD. For models 2 and 3, our results for ISIS-SCAD were far better

than with SIS-SCAD. For model 2, ISIS-SCAD had slightly better power than the SCAD implementation by Koh et al. (2011), but slightly lower coverage. For model 3, however, ISIS-SCAD produced approximately 25% better power and coverage than Koh et al.'s (2011) use of SCAD. However, for both models 2 and 3, Koh et al.'s (2011) use of the permutation test and stepwise selection procedures were a bit more successful than our ISIS-SCAD procedure.

4.1.4 Application of SIS and ISIS Variable Selection Procedures to the Actual Williams Rubber Data

For our simulations, we used a half-fraction of the 28 run, 24 factor Plackett-Burman design originally used in Williams's (1968) rubber data study. The original response data for the half-fraction is available in Lin's 1993 paper and is shown in Table 4.5. Note that factors 13 and 16 have been combined since their factor levels are identical in this half-fraction.

Table 4.5: Half-fraction of the Williams rubber data 28-run Plackett-Burman design, as suggested by Lin (1993), including the original response

Factors																							
1	2	3	4	5	6	7	8	9	10	11	12	13/16	14	15	17	18	19	20	21	22	23	24	Y
+	+	+	−	−	−	+	+	+	+	+	−	+	−	−	+	−	−	+	−	−	−	+	133
+	−	−	−	−	−	+	+	+	−	−	−	+	+	+	−	+	−	−	+	+	−	−	62
+	+	−	+	+	−	−	−	−	+	−	+	+	+	+	+	−	−	−	−	+	+	−	45
+	+	−	+	−	+	−	−	−	+	+	−	+	−	+	−	+	+	+	−	−	−	−	52
−	−	+	+	+	+	−	+	+	−	−	−	+	−	+	+	−	−	+	−	+	+	+	56
−	−	+	+	+	+	+	−	+	+	+	−	−	+	+	+	+	+	+	+	+	−	−	47
−	−	−	−	+	−	−	+	−	+	−	+	+	+	−	+	+	+	+	+	−	−	+	88
−	+	+	−	−	+	−	+	−	+	−	−	−	−	−	−	−	+	−	+	+	+	−	193
−	−	−	−	−	+	+	−	−	−	+	+	−	−	+	+	+	−	−	−	−	+	+	32
+	+	+	+	−	+	+	+	−	−	−	+	−	+	+	+	−	+	−	+	−	−	+	53
−	+	−	+	+	−	−	+	+	−	+	−	−	+	−	−	+	+	−	−	−	+	+	276
+	−	−	−	+	+	+	−	+	+	+	+	+	−	−	−	−	+	−	+	+	+	+	145
+	+	+	+	+	−	+	−	+	−	−	+	−	−	−	−	+	−	+	+	−	+	−	130
−	−	+	−	−	−	−	−	−	−	+	+	−	+	−	−	−	−	+	−	+	−	−	127

When Williams (1968) analyzed the original 28 run, 24 factor Plackett-Burman design, he identified factors 15, 20 and 17 as important, and factors 4, 22, 14 and 8 as moderate. After combining these results with other experimental results, he suggested that factors 15, 10, 20 and 4 be considered for additional study (Williams, 1968).

Lin (1993) used stepwise variable selection to analyze the half-fraction design he created (shown in Table 4.5), which selected factors 15, 12, 20, 4, 10 for the model. Factors 17, 22,

14 and 8 from Williams's (1968) original analysis were not identified, and factors 10 and 12 were found by Lin (2003) but not by Williams (1968). However, the set of factors does align with the set that was eventually suggested by Williams (1968) for additional study, with the addition of factor 12.

Abraham et al. (1999) later used an all-subsets approach to analyze the half-fraction design. Out of all the subsets of size three, the one with factors 1, 15 and 20 was identified as the best. The model with factors 1, 14, 15 and 20 was the best subset of size four, and the set of factors 1, 3, 14, 15 and 20 was the best subset of size five. Factors 15 and 20 align with the other analyses, and factor 14 was identified as a moderate effect in Williams (1968) original analysis; however, factors 1, 3 and 14 were not selected by Williams (1968) or Lin (1993) in their final models.

Li and Lin (2002) used stepwise variable selection followed by SCAD to analyze the half-fraction design. The initial stepwise procedure identified factors 15, 12, 20, 4, 10, 11, 7, 1, 14, 17 and 22, and then SCAD was applied. The final model selected by SCAD included factors 4, 12, 15 and 20. This is similar to the set selected by Williams (1968), except factor 12 was identified instead of factor 10. Factors 4, 15 and 20 were consistent with Williams's (1968) original results, and factor 12 was also included in Lin's (2003) stepwise procedure.

The results above are summarized in Table 4.6.

Table 4.6: Models recommended from analysis of the Williams (1968) rubber data

Design	Who (When)	Method	Model
complete P-B	Williams (1968)	original analysis	15 17 20 (4 8 14 22)
complete P-B	Williams (1968)	final recommendation	4 10 15 20
half-fraction SSD	Lin (1993)	stepwise	4 10 12 15 20
half-fraction SSD	Abraham et al. (1999)	all subsets (size 3)	1 15 20
half-fraction SSD	Abraham et al. (1999)	all subsets (size 4)	1 14 15 20
half-fraction SSD	Abraham et al. (1999)	all subsets (size 5)	1 3 14 15 20
half-fraction SSD	Li and Lin (2002)	stepwise-SCAD	4 12 15 20

We applied SIS-SCAD and ISIS-SCAD to the half-fraction design, and the results are shown in Table 4.7. Recall that these procedures require the specification of the maximum number of factors to return. We performed both selection procedures for the entire range of values less than n (i.e. from 1 to $n-1=13$). In every case except one (ISIS for $n-1=13$ factors), the selection process for both SIS and ISIS returned exactly the number of factors we had requested. The ISIS procedure for $n-1=13$ factors only returned 12 factors. Following SIS and ISIS by SCAD did not further reduce any of the models; however, it provided the coefficient estimates, which we used to calculate the R^2 and adjusted R^2 for each model.

Table 4.7: Models recommended by SIS-SCAD and ISIS-SCAD for the half-fraction Williams (1968) rubber data set

Selection Method	Max # Factors	# Factors Selected	Factors Selected	R^2	Adj R^2
SIS-SCAD	13	13	1 2 6 7 8 9 11 12 13/16 15 17 19 23	1.00	-
	12	12	1 2 6 7 8 9 12 13/16 15 17 19 23	0.94	0.28
	11	11	2 6 7 8 9 12 13/16 15 17 19 23	0.94	0.63
	10	10	2 6 7 8 9 13/16 15 17 19 23	0.93	0.71
	9	9	2 6 8 9 13/16 15 17 19 23	0.92	0.72
	8	8	2 6 8 13/16 15 17 19 23	0.89	0.72
	7	7	2 6 8 13/16 15 17 23	0.87	0.71
	6	6	2 6 8 15 17 23	0.84	0.70
	5	5	2 6 15 17 23	0.77	0.63
	4	4	2 15 17 23	0.77	0.67
	3	3	2 15 17	0.74	0.66
	2	2	15 17	0.69	0.64
	1	1	15	0.63	0.60
ISIS-SCAD	13	12	1 2 6 9 11 12 13/16 15 17 19 21 23	1.00	1.00
	12	12	2 6 8 9 11 12 13/16 15 17 19 21 23	0.99	0.83
	11	11	2 6 8 11 12 13/16 14 15 17 23 24	1.00	1.00
	10	10	1 2 6 8 11 12 14 15 17 23	1.00	0.98
	9	9	1 2 6 8 11 14 15 17 23	0.99	0.96
	8	8	1 2 6 12 14 15 17 23	0.95	0.86
	7	7	1 2 8 12 15 17 23	0.89	0.77
	6	6	1 2 12 15 17 23	0.88	0.77
	5	5	1 2 15 17 24	0.88	0.81
	4	4	8 12 15 17	0.79	0.69
	3	3	12 15 17	0.77	0.70
	2	2	12 15	0.74	0.69
	1	1	15	0.63	0.60

It appears that factor 15 alone explains 63% of the variation in response, and it was found by SIS and ISIS every time. Factor 20, however, was recommended in all the prior studies, but it was not found by SIS or ISIS. Factors 4 and 10 were recommended by several previous studies, including the original recommendation by Williams (1968); however, they were also not found by SIS and ISIS. Factor 12, which was found by Lin (1993) and Li and Lin (2002), was identified by ISIS as early as the 2-factor model. (It was not, however, included in the 5-factor or 9-factor ISIS models.) Factor 12 was also found by SIS, but not until the 11-factor model. Factor 17, which was identified as important in Williams's (1968) original analysis (but not recommended by him later on, or by anyone else), was found in all 2+ factor SIS models and all 3+ factor ISIS models. In general, it seems that the SIS and ISIS results are inconsistent with some of the prior results, with the exception of factor 15.

4.2 Marley and Woods Designs

4.2.1 Comparison of Forward Selection, Gauss-Dantzig Selector, and Model Averaging Variable Selection Procedures by Marley and Woods (2010)

In 2010, Marley and Woods performed a simulation study to compare forward selection, screening by Gauss-Dantzig selector, and model averaging approaches for Bayesian D-optimal and $E(s^2)$-optimal designs. They selected three Bayesian D-optimal SSDs and three $E(s^2)$-optimal SSDs. For each kind of optimality, there was one 22 factor, 18 run design, one 24 factor, 14 run design, and one 26 factor, 12 run design, as shown in Tables 4.8, 4.9, 4.10, 4.11, 4.12 and 4.13. This set of six designs provided a breadth of supersaturated-ness for the simulation study.

Table 4.8: Marley and Woods (2010) Bayesian D-optimal design with 22 factors and 18 runs

Factors																					
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
−	+	−	−	−	−	−	+	+	−	+	−	−	+	−	−	+	+	−	+	−	+
+	+	−	−	−	−	+	+	−	−	−	+	+	+	+	+	−	−	+	+	+	+
+	+	−	+	−	+	−	−	+	+	+	−	−	+	−	+	−	−	+	−	−	+
−	−	−	+	−	−	−	−	−	+	+	+	+	+	+	−	−	+	+	+	−	−
+	−	−	−	+	+	−	−	−	−	+	+	+	+	−	−	+	−	−	−	+	+
−	+	+	+	+	−	−	−	+	−	−	+	+	−	+	−	+	−	+	−	−	+
−	−	−	−	+	−	−	+	+	+	+	+	−	−	+	+	−	−	−	−	+	−
−	−	+	−	−	+	+	+	+	−	−	−	+	+	+	−	−	−	−	−	−	−
+	+	+	−	−	+	−	+	−	+	−	+	−	−	−	−	+	+	+	−	+	−
−	−	−	+	−	+	+	−	−	+	−	−	−	−	+	−	+	−	−	+	+	+
+	−	+	−	−	−	+	−	−	+	+	−	+	−	+	+	+	+	−	−	−	+
+	+	−	−	+	+	−	−	−	−	−	−	−	−	+	+	−	+	−	+	−	−
−	+	−	+	+	+	+	+	−	−	+	−	+	−	−	+	+	+	+	−	+	−
+	−	+	−	+	−	+	−	+	−	+	−	−	−	−	−	−	−	+	+	+	−
−	−	+	+	−	+	−	−	+	−	−	+	+	−	−	+	−	+	−	+	+	+
+	−	+	+	+	−	−	+	−	+	−	−	+	+	−	+	+	−	−	+	−	−
−	−	−	−	+	+	+	−	+	+	−	+	−	+	−	+	+	+	+	+	−	+
+	+	+	+	+	+	+	+	+	+	+	+	−	+	+	−	−	+	−	+	+	+

Table 4.9: Marley and Woods (2010) Bayesian D-optimal design with 24 factors and 14 runs

Factors																							
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
+	+	+	−	−	−	−	−	+	−	−	−	−	+	−	+	−	+	+	+	−	−	−	+
−	+	−	+	+	+	+	+	+	+	−	−	+	+	+	+	−	−	−	−	−	−	−	+
+	+	−	−	+	+	+	+	+	−	+	−	−	−	−	−	+	+	−	−	−	+	+	−
+	+	−	−	−	+	−	−	+	+	−	+	+	−	+	−	−	−	−	+	+	+	+	+
−	−	+	+	+	−	−	−	+	−	−	+	+	−	−	−	+	+	−	−	+	+	−	+
+	−	−	−	+	−	−	+	−	−	−	+	+	+	+	+	+	−	−	+	−	+	−	−
+	−	+	+	−	+	+	+	+	−	+	+	+	+	+	+	+	+	+	+	+	−	+	+
−	−	−	−	−	+	+	−	−	+	−	−	−	+	+	−	+	+	+	−	+	+	−	−
+	−	+	+	−	−	+	+	−	+	+	−	−	−	+	−	−	−	−	+	−	+	−	+
−	−	−	−	−	−	+	−	−	+	+	+	+	−	−	+	−	+	−	−	−	−	+	−
−	+	+	−	+	−	−	+	−	+	+	−	+	+	−	−	+	−	+	+	+	−	+	+
+	−	+	+	+	+	−	−	−	−	−	−	+	−	+	−	−	−	+	−	−	−	+	−
−	+	−	+	−	+	−	+	−	−	+	+	−	−	−	+	−	−	+	−	+	+	−	−
−	+	+	+	+	−	+	−	+	+	−	+	−	−	+	+	+	−	+	+	−	+	+	−

Table 4.10: Marley and Woods (2010) Bayesian D-optimal design with 26 factors and 12 runs

Factors																									
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
−	−	−	+	+	+	+	+	−	−	−	−	−	−	−	−	−	−	+	+	+	+	−	−	−	−
+	+	+	+	+	−	+	+	+	+	+	−	+	−	+	+	+	+	−	+	+	+	+	−	−	−
−	−	+	−	+	+	+	−	+	−	−	+	+	−	+	+	+	−	+	−	−	−	+	−	+	+
−	+	+	+	−	+	−	+	−	+	+	−	+	−	+	−	−	+	+	−	−	−	−	−	+	+
+	−	+	−	−	+	−	+	+	−	−	−	+	+	−	−	−	+	−	+	+	−	+	+	+	−
+	−	−	−	+	+	+	+	−	+	+	+	−	+	+	+	−	+	−	−	+	−	−	+	−	+
−	+	−	−	+	−	−	−	−	+	−	−	−	+	+	+	−	+	+	+	−	+	+	+	+	−
+	+	−	+	−	+	−	−	+	−	+	+	−	−	−	+	+	+	+	−	+	+	−	+	+	−
−	+	−	−	+	−	−	−	+	+	−	−	+	−	−	−	+	−	−	−	+	−	−	+	−	+
+	−	+	+	−	−	−	+	−	+	−	+	+	+	−	+	+	−	+	+	−	+	−	+	−	+
−	+	+	−	−	−	+	−	−	−	+	+	−	+	−	−	+	+	−	+	−	−	−	−	−	−
+	−	−	+	−	−	−	−	+	−	+	−	−	+	+	−	−	−	−	−	−	+	+	−	−	+

Table 4.11: Marley and Woods (2010) $E(s^2)$-optimal design with 22 factors and 18 runs

Factors																					
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
−	−	−	−	+	−	−	−	+	−	−	+	−	−	+	+	−	−	−	+	+	+
+	+	+	+	−	−	−	+	+	−	+	+	−	−	+	−	+	−	+	+	−	+
−	−	+	−	−	+	+	+	+	−	−	+	−	+	−	+	+	+	+	−	+	−
+	+	−	+	+	−	−	+	+	+	+	+	−	+	−	−	−	+	−	−	+	−
+	+	+	+	−	−	+	−	−	+	−	+	+	−	−	+	+	+	−	+	+	−
+	−	−	+	+	+	+	+	−	−	+	−	−	−	−	+	+	−	−	−	+	+
−	+	−	−	−	+	−	−	+	−	+	−	+	−	−	−	+	+	−	−	−	+
−	−	−	+	+	−	+	−	−	−	−	−	−	+	+	−	+	+	−	+	−	−
+	−	−	−	−	−	+	+	+	+	+	−	+	+	+	+	−	+	−	+	−	+
+	+	+	+	+	−	+	−	+	−	−	−	+	+	−	+	−	−	+	−	−	+
+	−	+	−	+	+	−	+	−	+	−	+	+	+	−	−	+	−	−	+	−	+
+	+	−	−	+	+	+	+	−	−	−	+	+	−	+	−	−	+	+	−	−	−
−	−	+	+	−	+	−	−	−	−	+	+	+	+	+	+	−	−	−	−	−	−
+	−	+	+	−	+	−	−	−	+	−	−	−	−	+	−	−	+	+	−	+	+
−	+	+	−	+	−	−	+	−	+	+	−	−	−	−	+	−	+	+	+	−	−
−	+	−	−	−	+	+	−	−	+	+	+	−	+	−	−	−	−	+	+	+	+
−	−	+	−	+	−	+	−	+	+	+	−	+	−	+	−	+	−	+	−	+	−
−	+	−	+	−	+	−	+	+	+	−	−	+	+	+	+	+	−	+	+	+	−

Table 4.12: Marley and Woods (2010) $E(s^2)$-optimal design with 24 factors and 14 runs

Factors																							
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
+	+	−	+	−	+	−	+	+	−	−	+	−	+	−	−	−	+	−	−	−	−	+	+
−	−	+	+	−	−	+	−	+	+	−	+	+	+	+	+	−	+	+	+	+	+	+	+
−	+	−	+	+	−	+	+	+	−	+	−	+	−	+	−	+	−	−	+	−	+	+	+
+	−	−	−	+	+	+	+	+	−	−	+	+	+	−	+	+	+	+	+	−	+	−	−
+	+	−	−	+	−	−	−	−	+	−	−	+	+	+	+	+	+	−	−	+	−	+	−
−	−	+	−	−	−	−	−	−	−	+	+	−	+	+	−	+	−	+	−	−	+	+	−
−	−	+	+	+	+	−	+	−	+	−	+	−	+	+	−	+	−	−	+	+	−	−	+
+	−	−	−	−	+	+	+	+	+	+	−	−	−	+	+	+	−	+	−	+	−	+	+
−	−	−	−	−	+	−	−	−	−	−	−	+	−	−	+	−	−	−	−	−	+	−	+
+	+	+	−	−	−	−	+	+	+	+	−	+	+	−	−	−	−	−	+	+	+	−	−
−	−	−	+	+	−	−	−	+	−	+	−	−	−	−	−	−	+	+	+	+	−	−	−
−	+	+	+	−	−	+	+	−	+	−	−	−	−	−	+	+	+	+	−	−	−	−	−
+	+	+	−	+	+	+	−	−	−	+	+	+	−	+	−	−	+	+	−	+	−	−	+
+	+	+	+	+	+	+	−	−	+	+	+	−	−	−	+	−	−	−	+	−	+	+	−

Table 4.13: Marley and Woods (2010) $E(s^2)$-optimal design with 26 factors and 12 runs

Factors																									
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
−	+	+	+	−	−	−	−	−	+	+	−	−	+	+	+	−	+	−	−	+	−	+	+	−	+
−	−	−	−	+	+	+	−	−	+	+	−	−	+	+	−	+	−	−	+	−	+	+	+	+	−
+	+	+	+	+	−	+	−	+	−	−	+	−	+	−	+	−	−	−	−	−	+	−	+	+	−
−	+	+	−	−	+	−	+	+	−	−	−	−	−	+	+	+	−	+	−	+	+	−	−	+	+
+	−	−	−	+	+	+	−	−	−	−	−	−	−	−	−	−	+	−	−	+	−	−	−	−	+
−	+	+	+	−	+	+	−	−	−	−	+	+	−	−	−	+	+	+	+	−	+	+	+	−	+
−	+	−	+	+	−	−	−	+	+	+	+	+	−	+	−	+	+	+	−	−	−	−	−	+	−
−	−	−	−	−	+	−	+	+	−	+	+	+	+	−	+	−	+	−	+	−	+	−	−	−	−
+	−	+	−	+	−	+	+	−	−	+	+	+	+	+	+	+	+	+	+	+	−	+	−	+	+
+	−	−	+	−	−	+	+	+	+	−	−	+	+	+	−	−	−	+	−	+	+	+	−	−	−
+	+	−	−	+	−	−	+	+	+	−	−	+	−	−	+	+	−	−	+	−	−	+	+	−	+
+	−	+	+	−	+	−	+	−	+	+	+	−	−	−	−	−	−	+	+	+	−	−	+	+	−

For each design, Marley and Woods (2010) examined four model scenarios in order to study the following characteristics of interest:

1. Effect sparsity: $c = 3$ factors, coefficients selected from a $N(\mu, 0.2)$ with $\mu = 5$
2. Intermediate complexity: $c = 4$ or $c = 5$ factors (chosen with equal probability), coefficients selected from a $N(\mu, 0.2)$ with $\mu = 4$
3. Large number of small effects: $c = 6$ factors, coefficients selected from a $N(\mu, 0.2)$ with $\mu = 3$
4. Large number of effects of mixed size: $c = 9$ factors, coefficients selected from $N(\mu, 0.2)$ distributions – one with $\mu = 10$, one with $\mu = 8$, one with $\mu = 5$, one with $\mu = 3$, and five with $\mu = 2$

For each of these four scenarios, Marley and Woods (2010) performed 10,000 iterations of selecting c factors randomly from the k possible factors in the model matrix $\mathbf{X}$. The coefficients were then sampled from the appropriate $N(\mu, 0.2)$ distribution(s) and randomly assigned to the c selected factors, with $\pm$ also being assigned randomly to the coefficients. Further, coefficients for the $k-c$ inactive factors were also randomly sampled from a $N(0, 0.2)$ distribution. Data were then generated from each model with random error distribution $N(0, 1)$, and analyzed by the three procedures: forward selection, Gauss-Dantzig selector, and model averaging. The analyses were compared in terms of:

1. Power (P) - Average percentage of active factors that were correctly identified as active. (Note: For scenario 4, the power was also calculated separately for the dominant effects (P(10,8)), moderate effects (P(5,3)), and small effects (P(2)).)
2. Type I Error Rate (T1) - Average percentage of inactive factors that were incorrectly identified as active
3. Coverage (C) - Percentage of simulations where a method identified all the active factors and possibly a few other factors
4. Mean (M) - Average number of factors identified as being active

The identification rate of the exact model was not examined as part of this study. It is possible that Marley and Woods (2010) were more concerned about avoiding type II errors than type I errors, and therefore would have favored power and coverage over the identification rate of the exact model.

As one might expect, all of the selection methods performed the best under scenario 1 (effect sparsity), with 100% power and coverage for both 22 factor, 18 run designs. Power and coverage fell to between 81% - 99% for the 24 factor, 14 run designs, and the 26 factor, 12 run designs only had power and coverage between 47% - 89%.

The selection methods also performed moderately well under scenario 2 (intermediate complexity). Power and coverage both dropped, however, for scenarios 3 and 4 (large number of small effects, and large number of mixed size effects), and this was especially evident in the 24 factor, 14 run and 26 factor, 12 run designs. Marley and Woods (2010) recommended that in general, the number of experimental runs for an SSD should be at least three times the number of active factors in order to ensure accurate variable selection results.

Also consistently evident was that as the ratio of factors to runs increased, variable selection capabilities got worse. This was true of all the scenarios and of all three variable selection methods, although the reduction in power and coverage that occurred as the scenarios increased in complexity became more and more pronounced as the ratio of factors to runs increased. One of Marley and Woods's (2010) recommendations for SSDs was that the overall ratio of factors to runs should be kept at less than two.

4.2.2 Comparison of SIS and ISIS Variable Selection Procedures

For our simulation study, we used a simplified version of the model selection process used by Marley and Woods (2010). We wanted to avoid model selection bias, so for each design, we randomly generated two models from each of the following five scenarios:

1. Effect sparsity: $c = 3$ factors, coefficients equal to $\mu = 5$
2. Intermediate complexity (a): $c = 4$ factors, coefficients equal to $\mu = 4$
3. Intermediate complexity (b): $c = 5$ factors, coefficients equal to $\mu = 4$
4. Large number of small effects: $c = 6$ factors, coefficients equal to $\mu = 3$
5. Large number of effects of mixed size: $c = 9$ factors, one coefficient equal to $\mu = 10$, one equal to $\mu = 8$, one equal to $\mu = 5$, one equal to $\mu = 3$, and five equal to $\mu = 2$

For each of the ten resulting models (two from each scenario), signs ($\pm$) for the coefficients were randomly assigned, and inactive factors were assigned coefficients of zero. Data (1000 iterations) were generated from each model with random error distribution $N(0, 1)$ and analyzed by SIS and ISIS, followed by SCAD down-selection / coefficient estimation procedures.

The SCAD tuning parameter for every case was chosen by BIC since that selection method yielded slightly better results in the Williams design simulations. Similarly to the Williams design models, the analyses were compared in terms of six criteria:

1. Identification rate (IR) - Percentage of simulations where a method identified exactly all the active factors, but no inactive factors
2. Coverage (C) - Percentage of simulations where a method identified all the active factors and possibly a few other factors
3. Power (P) - Average percentage of active factors that were correctly identified as active. (Note: For scenario 5, the power was also calculated separately for the dominant effects (P(10,8)), moderate effects (P(5,3)), and small effects (P(2)).)
4. Type I Error Rate (T1) - Average percentage of inactive factors that were incorrectly identified as active
5. Mean (M) - Average number of factors identified as being active
6. Variance (V) - Variance of the number of factors identified as being active

The statistical package R 2.14.1 was used to generate all of the models, data and analysis for this study. All of the R code is shown in the Appendix. The detailed results of the simulation study are shown in Tables 4.14, 4.15, 4.16, 4.17, 4.18 and 4.19.

For scenario 1 (effect sparsity with three factor coefficients equal to ± 5), ISIS-SCAD had coverage and power above 95% for all designs except design 3, which still had coverage near 90% and power near 93%. (Note that design 3 had 26 factors with only 12 runs.) The identification rate for all of the designs using ISIS-SCAD was between 20% to 25%. The SIS-SCAD method produced coverage and power of 92% or higher for all designs except design 3, which had coverage of only 64% and power of 82%. Clearly this design benefited from the additional iterative component of ISIS. The identification rates using SIS-SCAD were still near those of ISIS-SCAD, between 20% and 30%. For very sparse models with moderate to large effects, it appears that ISIS-SCAD (and to some extent SIS-SCAD) are reliable analysis methods.

For scenario 2 (intermediate complexity with four factor coefficients equal to ± 4), ISIS-SCAD had coverage and power of 94% or higher for all designs except design 6, which still had coverage near 84% and power near 93%. (Like design 3, design 6 had 26 factors with only 12 runs.) The identification rate for all of the designs using ISIS-SCAD was between 20% to 27%. The SIS-SCAD method produced coverage and power of 84% or higher for all designs. The identification rates using SIS-SCAD were still near those of ISIS-SCAD, between 19%

and 28%. This supports our conclusion from scenario 1 that for sparse models with moderate to large effects, ISIS-SCAD (and to some extent SIS-SCAD) are viable analysis methods.

For scenario 3 (intermediate complexity with five factor coefficients equal to ± 4), ISIS-SCAD had coverage and power of 84% or higher for designs 1, 3, 4 and 5. The identification rate for these designs using ISIS-SCAD was between 22% and 25%. Design 2 had coverage of only 17% and power of 65%, and design 6 had coverage of only 50% and power 72%. (These were two of the larger designs with fewer runs.) The identification rate for these two designs were only 5% and 12%, respectively. The SIS-SCAD method produced coverage and power of 90% or higher for designs 1, 4 and 5. The identification rates for these designs for SIS-SCAD were between 24% and 27%. Designs 2, 3 and 6 were less successful, with lower coverage, power and identification rates, although design 3 performed a bit better than 2 and 6. As we add more active factors, the ISIS-SCAD and SIS-SCAD methods appear to perform less consistently, particularly for designs with a large number of factors and small number of runs.

For scenario 4 (large number of factors with small effects, six factor coefficients equal to ± 3), ISIS-SCAD retained high power (96% or higher) and good coverage (82% or higher) for models 1, 4 and 5. The identification rate for these models remained at 20% or higher. Similarly, SIS-SCAD had very good power (95% or higher) and good coverage (74% or higher), with identification rates between 18% and 25%. For designs 2, 3 and 6, the coverage, power and identification rate slipped similarly to in scenario 3, with coverage below 45%. This supports our suspicion that as we add more active factors, the ISIS-SCAD and SIS-SCAD methods appear to perform less consistently, especially for designs with a large number of factors and small number of runs.

As we might expect from reviewing scenarios 3 and 4, scenario 5 did not perform strongly (large number of factors with varied effects - four moderate to dominant factors, one each with coefficients 10, 8, 5 and 3, plus five with coefficients equal to 2). For the 22 factor, 18 run designs (1 and 4), ISIS-SCAD and SIS-SCAD had coverage of 28% to 45%, and power near 88%. The identification rate was low at around 10%. However, the other four designs (2, 3, 5 and 6) – both of the 24 factor, 14 run designs, and both of the 26 factor, 12 run designs – had very poor coverage, often 0%. These designs had power hovering around 60%, indicating that only around two-thirds of the factors were typically found with these designs.

For scenario 5, Table 4.20 shows the power for the dominant, moderate, and small effects. As one might expect, the dominant effects were found over 80% of the time for all models, with many found over 95% of the time. The moderate effects were found between 51% and 98% of the time. The small effects were found around 80% of the time for designs 1 and 4, but only between 30% to 60% of the time for designs 2, 3, 5 and 6. ISIS-SCAD and SIS-SCAD, therefore, seem to be useful for identifying the strong effects, but they suffer from under-fitting when there are a combination of large and small effects in the model.

These results are consistent with those from Marley and Woods (2010). We saw that as model complexity increased, the power and coverage for all of the designs decreased, and this was especially pronounced in the designs with a higher ratio of factors to runs. We consistently saw also that for nearly all scenarios, the designs with a higher ratio of factors to runs performed worse than the designs with a lower ratio of factors to runs.

Our results from using SIS and ISIS were comparable, and sometimes better than those of the Gauss-Dantzig selector, which had performed the best in Marley and Woods's (2010) study. SIS and ISIS were able to retain approximately 10% - 20% better power than the Gauss-Dantzig selector, even through the most complex model scenario in the 26 factor, 12 run designs, although coverage was still nearly 0% for these designs. This means that SIS and ISIS were able to identify more of the active factors, but still not all of them. Note, though, that our simulation only studied two randomly generated models for each scenario, whereas Marley and Woods (2010) iterated through 10,000 randomly generated models for each scenario. Therefore, this probably warrants further investigation to make sure our results are not coincidental.

Table 4.14: Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the Bayesian D-optimal design in 22 factors and 18 runs

Scenario	Method	Trial 1						Trial 2						Average					
		IR	C	P	T1	M	V	IR	C	P	T1	M	V	IR	C	P	T1	M	V
1^{ab}	SIS	0	100	100	74	17.0	0.00	0	100	100	74	17.0	0.00	0	100	100	74	17.0	0.00
	SIS-SCAD	20	91	97	29	8.4	19.84	26	97	99	20	6.7	12.33	23	94	98	24	7.6	16.09
	ISIS	21	99	100	21	7.1	11.42	24	99	100	21	6.9	11.40	23	99	100	21	7.0	11.41
	ISIS-SCAD	22	98	99	20	6.8	10.34	24	99	100	19	6.7	10.05	23	99	99	20	6.7	10.19
2^{cd}	SIS	0	100	100	72	17.0	0.00	0	100	100	72	17.0	0.00	0	100	100	72	17.0	0.00
	SIS-SCAD	26	94	98	22	7.9	13.99	25	93	98	25	8.4	16.64	26	93	98	24	8.2	15.32
	ISIS	23	97	99	21	7.8	10.55	21	98	99	21	7.8	10.07	22	97	99	21	7.8	10.31
	ISIS-SCAD	24	97	99	20	7.6	9.42	22	97	99	20	7.6	9.15	23	97	99	20	7.6	9.28
3^{ef}	SIS	0	100	100	71	17.0	0.00	0	100	100	71	17.0	0.00	0	100	100	71	17.0	0.00
	SIS-SCAD	24	89	97	24	9.0	13.55	24	93	98	24	9.0	13.21	24	91	98	24	9.0	13.38
	ISIS	22	97	99	21	8.5	8.62	21	97	99	21	8.6	8.57	22	97	99	21	8.6	8.59
	ISIS-SCAD	23	96	99	19	8.3	7.56	21	97	99	20	8.4	7.74	22	97	99	20	8.3	7.65
4^{gh}	SIS	0	100	100	69	17.0	0.00	0	84	97	70	17.0	0.00	0	92	99	69	17.0	0.00
	SIS-SCAD	21	89	98	25	9.9	10.24	23	80	96	28	10.2	13.73	22	84	97	27	10.1	11.98
	ISIS	22	91	98	23	9.6	9.02	17	74	94	32	10.7	11.53	19	82	96	28	10.2	10.27
	ISIS-SCAD	23	91	98	22	9.4	8.24	17	72	94	30	10.5	10.74	20	82	96	26	10.0	9.49
5^{ij}	SIS	0	63	96	64	17.0	0.00	0	3	87	71	17.0	0.00	0	33	91	68	17.0	0.00
	SIS-SCAD	17	54	92	39	13.3	7.77	1	2	82	61	15.3	2.53	9	28	87	50	14.3	5.15
	ISIS	5	19	82	56	14.6	3.48	18	73	96	28	12.3	5.71	12	46	89	42	13.5	4.60
	ISIS-SCAD	5	19	81	55	14.5	3.51	19	72	96	27	12.1	5.34	12	45	88	41	13.3	4.43

[a]Scenario 1, Trial 1: $y = -5X_{13} - 5X_{14} + 5X_{17} + \epsilon$
[b]Scenario 1, Trial 2: $y = 5X_1 + 5X_{15} + 5X_{20} + \epsilon$
[c]Scenario 2, Trial 1: $y = 4X_4 - 4X_{13} + 4X_{14} + 4X_{17} + \epsilon$
[d]Scenario 2, Trial 2: $y = -4X_3 - 4X_7 - 4X_{21} - 4X_{22} + \epsilon$
[e]Scenario 3, Trial 1: $y = -4X_9 + 4X_{12} - 4X_{13} - 4X_{14} - 4X_{18} + \epsilon$
[f]Scenario 3, Trial 2: $y = 4X_4 + 4X_8 + 4X_{13} - 4X_{18} + 4X_{19} + \epsilon$
[g]Scenario 4, Trial 1: $y = -3X_6 - 3X_{12} - 3X_{13} + 3X_{17} + 3X_{18} - 3X_{19} + \epsilon$
[h]Scenario 4, Trial 2: $y = -3X_3 - 3X_{12} + 3X_{13} + 3X_{18} + 3X_{20} + 3X_{22} + \epsilon$
[i]Scenario 5, Trial 1: $y = -2X_4 - 3X_5 + 2X_6 + 10X_7 - 5X_{10} - 2X_{13} + 2X_{16} + 2X_{19} + 8X_{20} + \epsilon$
[j]Scenario 5, Trial 2: $y = 3X_2 - 10X_3 + 2X_6 - 8X_{13} + 5X_{15} - 2X_{16} + 2X_{17} - 2X_{18} - 2X_{20} + \epsilon$

Table 4.15: Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the Bayesian D-optimal design in 24 factors and 14 runs

Scenario	Method	Trial 1						Trial 2						Average					
		IR	C	P	T1	M	V	IR	C	P	T1	M	V	IR	C	P	T1	M	V
1^{ab}	SIS	0	100	100	48	13.0	0.00	0	100	100	48	13.0	0.00	0	100	100	48	13.0	0.00
	SIS-SCAD	29	96	98	14	5.9	7.73	18	89	96	20	7.2	9.16	24	92	97	17	6.5	8.45
	ISIS	21	94	98	15	6.2	5.81	20	98	99	15	6.0	5.24	21	96	98	15	6.1	5.52
	ISIS-SCAD	22	94	97	15	6.0	5.40	21	98	99	14	5.9	4.74	21	96	98	14	5.9	5.07
2^{cd}	SIS	0	100	100	45	13.0	0.00	0	100	100	45	13.0	0.00	0	100	100	45	13.0	0.00
	SIS-SCAD	15	73	92	23	8.3	8.26	24	94	98	16	7.1	6.79	19	84	95	19	7.7	7.52
	ISIS	20	94	98	14	6.8	4.54	18	96	99	14	6.8	4.47	19	95	98	14	6.8	4.50
	ISIS-SCAD	20	94	98	14	6.6	4.13	19	96	99	13	6.6	4.08	20	95	98	14	6.6	4.10
3^{ef}	SIS	0	1	80	47	13.0	0.00	0	23	84	46	13.0	0.00	0	12	82	47	13.0	0.00
	SIS-SCAD	0	1	61	38	10.3	2.15	6	20	76	37	10.8	5.07	3	11	68	37	10.5	3.61
	ISIS	7	24	61	33	9.3	3.46	3	11	70	41	11.2	3.89	5	17	66	37	10.2	3.67
	ISIS-SCAD	7	24	61	32	9.1	3.38	3	10	70	40	11.1	4.10	5	17	65	36	10.1	3.74
4^{gh}	SIS	0	33	89	43	13.0	0.00	0	0	67	50	13.0	0.00	0	16	78	46	13.0	0.00
	SIS-SCAD	10	29	83	32	10.7	5.20	0	0	54	36	9.8	2.92	5	14	68	34	10.2	4.06
	ISIS	10	41	81	27	9.8	3.88	1	5	60	33	9.5	2.16	6	23	71	30	9.6	3.02
	ISIS-SCAD	11	40	81	27	9.6	3.92	1	5	59	32	9.3	2.05	6	23	70	29	9.5	2.99
5^{ij}	SIS	0	0	52	55	13.0	0.00	0	70	97	29	13.0	0.00	0	35	74	42	13.0	0.00
	SIS-SCAD	0	0	47	47	11.2	1.61	10	39	88	20	10.9	1.13	5	19	67	33	11.1	1.37
	ISIS	0	0	50	43	10.9	1.38	0	1	63	36	11.2	1.26	0	0	57	40	11.1	1.32
	ISIS-SCAD	0	0	50	42	10.8	1.48	0	1	62	36	11.0	1.38	0	0	56	39	10.9	1.43

[a]Scenario 1, Trial 1: $y = 5X_{21} - 5X_{22} - 5X_{24} + \epsilon$
[b]Scenario 1, Trial 2: $y = -5X_1 + 5X_{15} + 5X_{16} + \epsilon$
[c]Scenario 2, Trial 1: $y = -4X_{12} - 4X_{13} - 4X_{16} - 4X_{19} + \epsilon$
[d]Scenario 2, Trial 2: $y = -4X_7 + 4X_9 + 4X_{17} + 4X_{21} + \epsilon$
[e]Scenario 3, Trial 1: $y = -4X_9 - 4X_{11} - 4X_{14} + 4X_{15} + 4X_{18} + \epsilon$
[f]Scenario 3, Trial 2: $y = -4X_4 + 4X_8 + 4X_9 + 4X_{15} - 4X_{20} + \epsilon$
[g]Scenario 4, Trial 1: $y = 3X_4 + 3X_7 + 3X_8 - 3X_9 - 3X_{20} + 3X_{24} + \epsilon$
[h]Scenario 4, Trial 2: $y = 3X_1 - 3X_6 + 3X_8 - 3X_9 + 3X_{10} - 3X_{20} + \epsilon$
[i]Scenario 5, Trial 1: $y = -8X_1 - 2X_3 - 2X_5 + 2X_7 - 5X_{10} - 2X_{12} - 2X_{16} + 3X_{17} + 10X_{20} + \epsilon$
[j]Scenario 5, Trial 2: $y = 5X_4 - 3X_5 - 2X_7 - 2X_{10} - 2X_{11} - 2X_{14} + 8X_{16} - 2X_{21} - 10X_{23} + \epsilon$

Table 4.16: Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the Bayesian D-optimal design in 26 factors and 12 runs

Scenario	Method	Trial 1						Trial 2						Average					
		IR	C	P	T1	M	V	IR	C	P	T1	M	V	IR	C	P	T1	M	V
1^{ab}	SIS	0	100	100	35	11.0	0.00	0	38	79	38	11.0	0.00	0	69	90	36	11.0	0.00
	SIS-SCAD	29	93	98	12	5.6	6.00	10	36	67	25	7.7	5.41	20	64	82	18	6.7	5.71
	ISIS	20	98	99	10	5.3	2.93	20	81	87	15	6.0	4.59	20	90	93	12	5.7	3.76
	ISIS-SCAD	20	98	99	10	5.2	2.73	20	81	87	14	5.9	4.36	20	90	93	12	5.6	3.55
2^{cd}	SIS	0	100	100	32	11.0	0.00	0	100	100	32	11.0	0.00	0	100	100	32	11.0	0.00
	SIS-SCAD	24	89	97	11	6.3	3.91	27	92	98	11	6.4	4.30	26	91	97	11	6.4	4.10
	ISIS	24	97	99	9	6.0	2.44	21	98	99	9	6.0	2.20	23	97	99	9	6.0	2.32
	ISIS-SCAD	24	97	99	9	5.9	2.30	21	98	99	9	5.9	1.98	23	97	99	9	5.9	2.14
3^{ef}	SIS	0	11	82	33	11.0	0.00	0	100	100	29	11.0	0.00	0	55	91	31	11.0	0.00
	SIS-SCAD	2	8	70	27	9.2	1.41	34	99	100	9	6.8	3.16	18	53	85	18	8.0	2.28
	ISIS	27	80	94	10	6.9	2.43	22	89	97	10	7.0	2.35	24	84	96	10	7.0	2.39
	ISIS-SCAD	27	80	94	10	6.8	2.29	22	89	97	10	7.0	2.31	25	84	96	10	6.9	2.30
4^{gh}	SIS	0	100	100	32	11.0	0.00	0	0	67	35	11.0	0.00	0	50	83	33	11.0	0.00
	SIS-SCAD	22	88	96	12	6.4	3.70	0	0	50	31	9.1	0.86	11	44	73	21	7.8	2.28
	ISIS	13	49	74	20	7.4	4.01	0	0	49	29	8.8	1.17	7	25	61	25	8.1	2.59
	ISIS-SCAD	14	49	74	20	7.3	3.95	0	0	48	29	8.7	1.21	7	25	61	24	8.0	2.58
5^{ij}	SIS	0	0	79	23	11.0	0.00	0	0	46	40	11.0	0.00	0	0	63	32	11.0	0.00
	SIS-SCAD	0	0	77	19	10.2	0.25	0	0	40	33	9.2	0.88	0	0	59	26	9.7	0.56
	ISIS	0	0	70	18	9.3	0.76	0	0	38	29	8.4	1.26	0	0	54	23	8.8	1.01
	ISIS-SCAD	0	0	70	18	9.2	0.76	0	0	38	29	8.3	1.29	0	0	54	23	8.7	1.02

[a]Scenario 1, Trial 1: $y = -5X_1 - 5X_{20} - 5X_{23} + \epsilon$
[b]Scenario 1, Trial 2: $y = -5X_5 - 5X_{14} + 5X_{15} + \epsilon$
[c]Scenario 2, Trial 1: $y = 4X_2 + 4X_3 + 4X_{11} - 4X_{22} + \epsilon$
[d]Scenario 2, Trial 2: $y = 4X_1 + 4X_6 + 4X_{18} - 4X_{20} + \epsilon$
[e]Scenario 3, Trial 1: $y = 4X_3 - 4X_4 + 4X_{12} - 4X_{16} + 4X_{18} + \epsilon$
[f]Scenario 3, Trial 2: $y = 4X_1 - 4X_3 + 4X_{14} + 4X_{15} - 4X_{25} + \epsilon$
[g]Scenario 4, Trial 1: $y = -3X_6 - 3X_{12} - 3X_{13} + 3X_{17} + 3X_{18} - 3X_{19} + \epsilon$
[h]Scenario 4, Trial 2: $y = 3X_6 + 3X_{11} - 3X_{18} + 3X_{19} + 3X_{23} + 3X_{26} + \epsilon$
[i]Scenario 5, Trial 1: $y = -2X_2 + 3X_5 + 2X_8 + 2X_{10} + 2X_{13} + 2X_{14} - 8X_{17} - 5X_{23} - 10X_{25} + \epsilon$
[j]Scenario 5, Trial 2: $y = 2X_8 + 8X_{11} + 10X_{12} - 3X_{13} - 2X_{14} + 5X_{15} - 2X_{21} - 2X_{22} + 2X_{24} + \epsilon$

Table 4.17: Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the $E(s^2)$-optimal design in 22 factors and 18 runs

Scenario	Method	Trial 1						Trial 2						Average					
		IR	C	P	T1	M	V	IR	C	P	T1	M	V	IR	C	P	T1	M	V
1^{ab}	SIS	0	100	100	74	17.0	0.00	0	100	100	74	17.0	0.00	0	100	100	74	17.0	0.00
	SIS-SCAD	25	96	98	21	7.0	15.62	25	98	99	23	7.4	17.37	25	97	99	22	7.2	16.49
	ISIS	21	98	99	22	7.1	11.72	21	98	99	22	7.2	11.37	21	98	99	22	7.1	11.54
	ISIS-SCAD	21	98	99	20	6.8	10.63	22	98	99	21	7.0	10.01	21	98	99	21	6.9	10.32
2^{cd}	SIS	0	100	100	72	17.0	0.00	0	100	100	72	17.0	0.00	0	100	100	72	17.0	0.00
	SIS-SCAD	23	97	99	22	8.0	11.27	24	96	99	22	7.9	11.39	24	97	99	22	7.9	11.33
	ISIS	22	97	99	22	8.0	11.14	23	97	99	21	7.8	10.82	22	97	99	22	7.9	10.98
	ISIS-SCAD	22	97	99	21	7.7	9.99	24	97	99	20	7.5	9.49	23	97	99	20	7.6	9.74
3^{ef}	SIS	0	100	100	71	17.0	0.00	0	100	100	71	17.0	0.00	0	100	100	71	17.0	0.00
	SIS-SCAD	24	97	99	20	8.4	9.94	29	97	99	20	8.3	10.65	27	97	99	20	8.4	10.29
	ISIS	22	96	99	22	8.7	9.39	24	97	99	21	8.6	9.32	23	96	99	22	8.6	9.36
	ISIS-SCAD	23	95	99	21	8.5	8.60	25	96	99	20	8.3	8.23	24	96	99	20	8.4	8.41
4^{gh}	SIS	0	100	100	69	17.0	0.00	0	100	100	69	17.0	0.00	0	100	100	69	17.0	0.00
	SIS-SCAD	27	96	99	18	8.9	7.58	22	85	97	25	9.9	11.81	25	91	98	22	9.4	9.70
	ISIS	24	95	99	21	9.3	7.80	22	90	98	24	9.8	10.21	23	93	98	23	9.5	9.01
	ISIS-SCAD	25	95	99	20	9.1	7.14	23	90	98	23	9.6	9.56	24	92	98	21	9.3	8.35
5^{ij}	SIS	0	34	93	67	17.0	0.00	0	97	100	62	17.0	0.00	0	65	96	64	17.0	0.00
	SIS-SCAD	9	31	83	49	13.8	5.59	8	51	93	42	13.9	6.18	9	41	88	45	13.9	5.89
	ISIS	10	38	88	45	13.8	6.06	8	44	90	43	13.7	5.31	9	41	89	44	13.7	5.68
	ISIS-SCAD	10	37	87	44	13.6	6.10	9	42	89	41	13.4	5.59	10	39	88	43	13.5	5.84

[a]Scenario 1, Trial 1: $y = 5X_6 - 5X_{10} - 5X_{19} + \epsilon$
[b]Scenario 1, Trial 2: $y = -5X_1 + 5X_{15} + 5X_{20} + \epsilon$
[c]Scenario 2, Trial 1: $y = 4X_1 + 4X_3 + 4X_{12} - 4X_{18} + \epsilon$
[d]Scenario 2, Trial 2: $y = 4X_{14} - 4X_{15} - 4X_{17} + 4X_{21} + \epsilon$
[e]Scenario 3, Trial 1: $y = 4X_2 - 4X_3 - 4X_5 - 4X_{17} - 4X_{19} + \epsilon$
[f]Scenario 3, Trial 2: $y = -4X_3 + 4X_4 - 4X_{16} - 4X_{18} + 4X_{20} + \epsilon$
[g]Scenario 4, Trial 1: $y = -3X_3 - 3X_9 - 3X_{12} + 3X_{15} + 3X_{18} + 3X_{19} + \epsilon$
[h]Scenario 4, Trial 2: $y = 3X_2 + 3X_7 + 3X_8 + 3X_9 - 3X_{17} - 3X_{21} + \epsilon$
[i]Scenario 5, Trial 1: $y = 2X_1 - 2X_3 + 3X_6 + 10X_8 + 2X_9 - 5X_{11} + 2X_{13} - 8X_{15} - 2X_{21} + \epsilon$
[j]Scenario 5, Trial 2: $y = 3X_1 - 5X_2 - 2X_3 + 2X_4 - 8X_8 + 2X_{10} - 10X_{16} - 2X_{17} + 2X_{18} + \epsilon$

Table 4.18: Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the $E(s^2)$-optimal design in 24 factors and 14 runs

Scenario	Method	Trial 1						Trial 2						Average					
		IR	C	P	T1	M	V	IR	C	P	T1	M	V	IR	C	P	T1	M	V
1^{ab}	SIS	0	100	100	48	13.0	0.00	0	100	100	48	13.0	0.00	0	100	100	48	13.0	0.00
	SIS-SCAD	27	95	98	14	5.9	6.99	22	95	98	17	6.4	7.96	25	95	98	15	6.2	7.48
	ISIS	24	99	99	14	5.8	5.17	21	99	100	14	6.0	5.16	22	99	100	14	5.9	5.16
	ISIS-SCAD	24	98	99	13	5.7	4.62	21	99	100	14	5.8	4.75	23	98	99	13	5.8	4.68
2^{cd}	SIS	0	100	100	45	13.0	0.00	0	100	100	45	13.0	0.00	0	100	100	45	13.0	0.00
	SIS-SCAD	22	93	98	15	7.0	6.16	19	81	94	21	8.0	8.47	20	87	96	18	7.5	7.31
	ISIS	19	92	98	15	7.0	4.77	22	97	99	14	6.7	4.37	20	95	98	14	6.8	4.57
	ISIS-SCAD	19	92	97	15	6.8	4.48	23	97	99	13	6.5	3.95	21	94	98	14	6.7	4.22
3^{ef}	SIS	0	100	100	42	13.0	0.00	0	100	100	42	13.0	0.00	0	100	100	42	13.0	0.00
	SIS-SCAD	30	100	100	12	7.2	4.78	24	88	97	16	7.9	6.19	27	94	99	14	7.6	5.49
	ISIS	21	96	99	13	7.4	3.48	25	93	98	13	7.4	4.27	23	95	99	13	7.4	3.88
	ISIS-SCAD	22	96	99	12	7.3	3.20	26	93	98	13	7.3	3.88	24	94	99	12	7.3	3.54
4^{gh}	SIS	0	100	100	39	13.0	0.00	0	100	100	39	13.0	0.00	0	100	100	39	13.0	0.00
	SIS-SCAD	14	68	94	21	9.4	4.16	22	79	96	17	8.7	4.37	18	74	95	19	9.1	4.26
	ISIS	23	95	99	12	8.0	2.52	23	92	98	12	8.1	2.74	23	94	99	12	8.1	2.63
	ISIS-SCAD	25	95	99	11	7.9	2.37	24	92	98	12	8.0	2.56	24	94	98	11	8.0	2.47
5^{ij}	SIS	0	0	69	45	13.0	0.00	0	0	78	40	13.0	0.00	0	0	73	43	13.0	0.00
	SIS-SCAD	0	0	62	39	11.4	0.70	0	0	75	35	12.0	0.72	0	0	69	37	11.7	0.71
	ISIS	0	2	59	37	10.9	1.20	0	0	73	28	10.8	1.58	0	1	66	33	10.8	1.39
	ISIS-SCAD	0	2	59	37	10.8	1.23	0	0	72	28	10.7	1.56	0	1	65	32	10.7	1.39

[a]Scenario 1, Trial 1: $y = 5X_{12} - 5X_{22} + 5X_{23} + \epsilon$
[b]Scenario 1, Trial 2: $y = 5X_3 + 5X_5 - 5X_{23} + \epsilon$
[c]Scenario 2, Trial 1: $y = -4X_2 + 4X_7 + 4X_{14} + 4X_{22} + \epsilon$
[d]Scenario 2, Trial 2: $y = 4X_2 + 4X_5 + 4X_{11} - 4X_{24} + \epsilon$
[e]Scenario 3, Trial 1: $y = -4X_1 - 4X_5 - 4X_{16} - 4X_{17} - 4X_{19} + \epsilon$
[f]Scenario 3, Trial 2: $y = -4X_{10} + 4X_{12} + 4X_{16} - 4X_{17} + 4X_{24} + \epsilon$
[g]Scenario 4, Trial 1: $y = -3X_1 - 3X_2 - 3X_3 - 3X_8 - 3X_{12} - 3X_{20} + \epsilon$
[h]Scenario 4, Trial 2: $y = -3X_4 + 3X_{12} + 3X_{17} - 3X_{18} - 3X_{21} + 3X_{23} + \epsilon$
[i]Scenario 5, Trial 1: $y = 5X_2 - 10X_4 - 3X_5 - 8X_{12} - 2X_{13} + 2X_{14} + 2X_{15} + 2X_{16} - 2X_{22} + \epsilon$
[j]Scenario 5, Trial 2: $y = -8X_1 - 3X_4 - 2X_8 - 5X_9 - 10X_{11} - 2X_{14} + 2X_{15} + 2X_{17} - 2X_{23} + \epsilon$

Table 4.19: Results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for the $E(s^2)$-optimal design in 26 factors and 12 runs

Scenario	Method	Trial 1						Trial 2						Average					
		IR	C	P	T1	M	V	IR	C	P	T1	M	V	IR	C	P	T1	M	V
1^{ab}	SIS	0	100	100	35	11.0	0.00	0	100	100	35	11.0	0.00	0	100	100	35	11.0	0.00
	SIS-SCAD	27	98	99	10	5.3	4.15	32	99	100	9	5.1	4.39	30	99	99	10	5.2	4.27
	ISIS	24	100	100	9	5.1	2.78	25	100	100	9	5.1	2.61	25	100	100	9	5.1	2.69
	ISIS-SCAD	24	100	100	9	5.1	2.56	25	100	100	9	5.0	2.43	25	100	100	9	5.0	2.49
2^{cd}	SIS	0	99	100	32	11.0	0.00	0	79	95	33	11.0	0.00	0	89	97	32	11.0	0.00
	SIS-SCAD	31	97	99	9	6.0	3.33	25	72	91	15	6.9	5.43	28	84	95	12	6.4	4.38
	ISIS	25	76	90	13	6.4	3.72	28	92	97	9	5.8	2.51	27	84	94	11	6.1	3.11
	ISIS-SCAD	25	76	90	13	6.4	3.57	28	92	97	8	5.7	2.37	27	84	93	11	6.0	2.97
3^{ef}	SIS	0	0	80	29	10.0	0.00	0	0	58	34	10.0	0.00	0	0	69	31	10.0	0.00
	SIS-SCAD	0	0	66	22	8.0	1.51	0	0	54	29	8.8	1.08	0	0	60	26	8.4	1.29
	ISIS	21	93	98	9	6.8	1.90	3	7	46	30	8.6	2.04	12	50	72	20	7.7	1.97
	ISIS-SCAD	22	93	98	9	6.8	1.84	3	7	46	30	8.6	2.08	12	50	72	19	7.7	1.96
4^{gh}	SIS	0	0	83	25	10.0	0.00	0	0	46	36	10.0	0.00	0	0	65	31	10.0	0.00
	SIS-SCAD	0	0	69	22	8.6	1.17	0	0	33	25	7.1	3.35	0	0	51	24	7.8	2.26
	ISIS	16	58	85	13	7.7	1.32	0	0	30	28	7.4	1.80	8	29	58	20	7.6	1.56
	ISIS-SCAD	16	58	85	13	7.7	1.28	0	0	30	28	7.3	1.78	8	29	58	20	7.5	1.53
5^{ij}	SIS	0	0	55	30	10.0	0.00	0	0	67	24	10.0	0.00	0	0	61	27	10.0	0.00
	SIS-SCAD	0	0	54	27	9.4	0.70	0	0	56	22	8.8	0.33	0	0	55	24	9.1	0.51
	ISIS	0	0	52	25	8.9	1.00	0	0	62	23	9.5	0.67	0	0	57	24	9.2	0.83
	ISIS-SCAD	0	0	52	25	8.9	1.04	0	0	62	23	9.4	0.73	0	0	57	24	9.2	0.88

[a]Scenario 1, Trial 1: $y = 5X_6 + 5X_{17} + 5X_{23} + \epsilon$
[b]Scenario 1, Trial 2: $y = 5X_{12} + 5X_{13} - 5X_{16} + \epsilon$
[c]Scenario 2, Trial 1: $y = 4X_5 - 4X_{16} + 4X_{20} + 4X_{26} + \epsilon$
[d]Scenario 2, Trial 2: $y = -4X_3 - 4X_{22} + 4X_{23} - 4X_{24} + \epsilon$
[e]Scenario 3, Trial 1: $y = -4X_4 - 4X_{11} - 4X_{17} - 4X_{20} - 4X_{23} + \epsilon$
[f]Scenario 3, Trial 2: $y = 4X_3 - 4X_7 - 4X_{16} + 4X_{20} - 4X_{21} + \epsilon$
[g]Scenario 4, Trial 1: $y = 3X_7 - 3X_9 - 3X_{10} + 3X_{17} - 3X_{23} + 3X_{26} + \epsilon$
[h]Scenario 4, Trial 2: $y = 3X_1 - 3X_8 + 3X_9 + 3X_{11} - 3X_{12} - 3X_{17} + \epsilon$
[i]Scenario 5, Trial 1: $y = 10X_3 + 2X_6 - 3X_{11} + 2X_{18} - 2X_{19} + 2X_{20} + 5X_{23} + 2X_{24} + 8X_{26} + \epsilon$
[j]Scenario 5, Trial 2: $y = -2X_1 + 5X_2 - 8X_4 + 2X_8 - 2X_{11} - 3X_{13} + 10X_{15} + 2X_{17} + 2X_{26} + \epsilon$

Table 4.20: Power results of our simulation comparison of SIS, SIS-SCAD, ISIS and ISIS-SCAD variable selection methods for all six Marley and Woods (2010) designs, scenario 5 only (large number of effects of mixed size)

Design	Method	Trial 1				Trial 2				Average			
		P	P(10,8)	P(5,3)	P(2)	P	P(10,8)	P(5,3)	P(2)	P	P(10,8)	P(5,3)	P(2)
1^{ab}	SIS	96	100	100	93	87	100	100	76	91	100	100	84
	SIS-SCAD	92	100	99	86	82	96	97	70	87	98	98	78
	ISIS	82	99	96	68	96	99	98	94	89	99	97	81
	ISIS-SCAD	81	99	96	68	96	99	98	93	88	99	97	81
2^{cd}	SIS	52	95	79	24	97	100	100	94	74	98	90	59
	SIS-SCAD	47	91	68	21	88	100	92	82	67	96	80	51
	ISIS	50	80	52	38	63	94	59	53	57	87	55	45
	ISIS-SCAD	50	79	51	37	62	92	58	52	56	86	55	45
3^{ef}	SIS	79	100	58	78	46	100	100	3	63	100	79	41
	SIS-SCAD	77	99	56	76	40	99	76	3	59	99	66	39
	ISIS	70	98	74	57	38	99	53	7	54	99	64	32
	ISIS-SCAD	70	98	74	56	38	99	53	7	54	99	63	32
4^{gh}	SIS	93	100	100	87	100	100	100	99	96	100	100	93
	SIS-SCAD	83	97	88	76	93	96	94	92	88	97	91	84
	ISIS	88	98	98	80	90	98	92	86	89	98	95	83
	ISIS-SCAD	87	98	98	79	89	97	92	85	88	97	95	82
5^{ij}	SIS	69	100	100	44	78	100	50	80	73	100	75	62
	SIS-SCAD	62	94	93	37	75	100	45	77	69	97	69	57
	ISIS	59	96	68	41	73	100	91	54	66	98	80	48
	ISIS-SCAD	59	95	68	40	72	99	91	54	65	97	79	47
6^{kl}	SIS	55	100	85	25	67	100	50	60	61	100	67	43
	SIS-SCAD	54	97	85	25	56	66	36	60	55	81	61	43
	ISIS	52	99	56	32	62	99	47	53	57	99	51	43
	ISIS-SCAD	52	99	56	32	62	99	47	53	57	99	51	43

[a]Design 1, Trial 1: $y = -2X_4 - 3X_5 + 2X_6 + 10X_7 - 5X_{10} - 2X_{13} + 2X_{16} + 2X_{19} + 8X_{20} + \epsilon$
[b]Design 1, Trial 2: $y = 3X_2 - 10X_3 + 2X_6 - 8X_{13} + 5X_{15} - 2X_{16} + 2X_{17} - 2X_{18} - 2X_{20} + \epsilon$
[c]Design 2, Trial 1: $y = -8X_1 - 2X_3 - 2X_5 + 2X_7 - 5X_{10} - 2X_{12} - 2X_{16} + 3X_{17} + 10X_{20} + \epsilon$
[d]Design 2, Trial 2: $y = 5X_4 - 3X_5 - 2X_7 - 2X_{10} - 2X_{11} - 2X_{14} + 8X_{16} - 2X_{21} - 10X_{23} + \epsilon$
[e]Design 3, Trial 1: $y = -2X_2 + 3X_5 + 2X_8 + 2X_{10} + 2X_{13} + 2X_{14} - 8X_{17} - 5X_{23} - 10X_{25} + \epsilon$
[f]Design 3, Trial 2: $y = 2X_8 + 8X_{11} + 10X_{12} - 3X_{13} - 2X_{14} + 5X_{15} - 2X_{21} - 2X_{22} + 2X_{24} + \epsilon$
[g]Design 4, Trial 1: $y = 2X_1 - 2X_3 + 3X_6 + 10X_8 + 2X_9 - 5X_{11} + 2X_{13} - 8X_{15} - 2X_{21} + \epsilon$
[h]Design 4, Trial 2: $y = 3X_1 - 5X_2 - 2X_3 + 2X_4 - 8X_8 + 2X_{10} - 10X_{16} - 2X_{17} + 2X_{18} + \epsilon$
[i]Design 5, Trial 1: $y = 5X_2 - 10X_4 - 3X_5 - 8X_{12} - 2X_{13} + 2X_{14} + 2X_{15} + 2X_{16} - 2X_{22} + \epsilon$
[j]Design 5, Trial 2: $y = -8X_1 - 3X_4 - 2X_8 - 5X_9 - 10X_{11} - 2X_{14} + 2X_{15} + 2X_{17} - 2X_{23} + \epsilon$
[k]Design 6, Trial 1: $y = 10X_3 + 2X_6 - 3X_{11} + 2X_{18} - 2X_{19} + 2X_{20} + 5X_{23} + 2X_{24} + 8X_{26} + \epsilon$
[l]Design 6, Trial 2: $y = -2X_1 + 5X_2 - 8X_4 + 2X_8 - 2X_{11} - 3X_{13} + 10X_{15} + 2X_{17} + 2X_{26} + \epsilon$

Chapter 5

Conclusion

5.1 Concluding Remarks

Our simulations indicated that ISIS-SCAD and SIS-SCAD methods of variable selection perform well when the true model is very sparse with moderate to large effects. If the number of effects becomes large (5 or greater), both ISIS-SCAD and SIS-SCAD have difficulties in detecting all of the active effects. Also, when there is a mixture of different sized effects, both ISIS-SCAD and SIS-SCAD can successfully detect the large effects, but typically are unsuccessful (or at best, inconsistent) in their detection of the smaller effects. We support Marley and Woods's (2010) recommendation that there should be at least three times as many experimental runs as active factors.

Further, ISIS-SCAD and SIS-SCAD are most effective when applied to designs where the number of factors does not exceed the number of runs by a large multiple. When the ratio of factors to runs approaches or exceeds two, ISIS-SCAD and SIS-SCAD begin to produce inconsistent results. This is consistent with the conclusions of Marley and Woods (2010).

For future implementations of SIS-SCAD and ISIS-SCAD, the BIC selection criterion is suggested for choosing the SCAD tuning parameter, λ. This selection method yielded slightly better results than the AIC method and is consistent with the work of Fan and Lv (2008).

5.2 Further Research

Further examination of the 'SIS' package code is suggested. For some designs, particular models were problematic, and even that was not consistent across runs. Some random error combinations allowed the model to be analyzed, and other random error combinations for the same model caused the analysis to fail. Some models seemed more prone to this kind of problem, such as model (4.1) with the Lin (1993) design from the Williams rubber data, and almost any model in the 26 factor, 12 run $E(s^2)$-optimal Marley and Woods design (2010). Further examination into the correlation structure of these designs may be appropriate.

Additional research and simulations can also be done with applications of ISIS-SCAD and SIS-SCAD to highly supersaturated designs (i.e. where the ratio of factors to runs approaches or exceeds two). These designs were the least successful in our studies. Reviewing the correlation structure of the simulated designs may provide additional insight into what makes ISIS-SCAD or SIS-SCAD more or less successful, or a different analysis approach altogether may need to be devised and tested.

Bibliography

[1] Abraham, B., Chipman, H. and Vijayan, K. (1999). Some risks in the construction and analysis of supersaturated designs. *Technometrics*, *41*, 135-141.

[2] Booth, K. H. V. and Cox, D. R. (1962). Some systematic supersaturated designs. *Technometrics*, *4*, 489-495.

[3] Candes, E. and Tao, T. (2007). The Dantzig selector: statistical estimation when p is much larger than n. *The Annals of Statistics*, *35*, 2313-2351.

[4] Chipman, H., Hamada, M. and Wu, C. F. J. (1997). A Bayesian variable-selection approach for analyzing designed experiments with complex aliasing. *Technometrics*, *39*, 372-381.

[5] Fan, J., Feng, Y., Samworth, R. and Wu, Y. (2010, Sept. 6). Package 'SIS': Sure independence screening. Retrieved from http://cran.r-project.org/web/packages/SIS/index.html

[6] Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, *96*, 1348-1360.

[7] Fan, J. and Lv, J. (2008). Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society*, *70*, 849-911.

[8] Gilmour, S. G. (2006). Factor screening via supersaturated designs. In A. Dean and S. Lewis (Eds.), *Screening: Methods for experimentation in industry, drug discovery, and genetics* (pp. 169-190). New York, NY: Springer.

[9] Holcomb, D. R. and Carlyle, W. M. (2002). Some notes on the construction and evaluation of supersaturated designs. *Quality and Reliability Engineering International*, *18*, 299-304.

[10] Holcomb, D. R., Montgomery, D. C. and Carlyle, W. M. (2003). Analysis of supersaturated designs. *Journal of Quality Technology*, *35*, 13-27.

[11] Jones, B., Lin, D. K. J., and Nachtsheim, C. J. (2008). Bayesian D-optimal supersaturated designs. *Journal of Statistical Planning and Inference*, *138*, 86-92.

[12] Koh, W. Y., Eskridge, K. M. and Wang, D. (2011). The effects of nonnormality on the analysis of supersaturated designs: a comparison of stepwise, SCAD and permutation test methods. *Journal of Statistical Computation and Simulation*, 1-9. doi:10.1080/00949655.2011.621953

[13] Li, R. and Lin, D. K. J. (2002). Data analysis in supersaturated designs. *Statistics and Probability Letters*, *59*, 135-144.

[14] Li, W. W. and Wu, C. F. J. (1997). Columnwise-pairwise algorithms with applications to the construction of supersaturated designs. *Technometrics*, *39*, 171-179.

[15] Lin, D. K. J. (1993). A new class of supersaturated designs. *Technometrics*, *35*, 28-31.

[16] Lin, D. K. J. (1995). Generating systematic supersaturated designs. *Technometrics*, *37*, 213-225.

[17] Lu, X., Hu, W., and Zheng, Y. (2003). A systematical procedure in the construction of multi-level supersaturated design. *Journal of Statistical Planning and Inference*, *115*, 287-310.

[18] Marley, C. J. and Woods, D. C. (2010). A comparison of design and model selection methods for supersaturated experiments. *Computational Statistics and Data Analysis*, *54*, 3158-3167.

[19] Montgomery, D. C. (2009). *Design and analysis of experiments.* Hoboken, NJ: John Wiley & Sons.

[20] Nguyen, N. K. (1996). An algorithmic approach to constructing supersaturated designs. *Technometrics*, *38*, 69-73.

[21] Phoa, F. K. H., Pan, Y. H. and Xu, H. (2009). Analysis of supersaturated designs via the Dantzig selector. *Journal of Statistical Planning and Inference*, *139*, 2362-2372.

[22] Plackett, R. L. and Burman, J. P. (1946). The design of optimum multifactorial experiments. *Biometrika*, *33*, 305-325.

[23] Satterthwaite, F. E. (1959). Random balance experimentation. *Technometrics*, *1*, 111-137.

[24] Williams, K. R. (1968). Designed experiments. *Rubber Age*, *100*, 65-71.

[25] Wu, C. F. J. (1993). Construction of supersaturated designs through partially aliased interactions. *Biometrika*, *80*, 661-669.

[26] Wu, C. F. J. and Hamada, M. S. (2009). *Experiments: Planning, analysis, and optimization.* Hoboken, NJ: John Wiley & Sons.

[27] Yamada, S. and Lin, D. K. J. (1999). Three-level supersaturated designs. *Statistics and Probability Letters*, *45*, 31-39.

Appendix A

R Code - Williams Rubber Data Design Simulations

```
#Load package 'SIS' first
library(SIS)

#Half-fraction of Plackett-Burman design from Lin (1993)
r1 = c(1, 1, 1, -1, -1, -1, 1, 1, 1, 1, 1, -1, 1, -1, -1, 1, 1, -1, -1, 1, -1, -1, -1, 1)
r2 = c(1, -1, -1, -1, -1, -1, 1, 1, 1, -1, -1, -1, 1, 1, 1, 1, -1, 1, -1, -1, 1, 1, -1, -1)
r3 = c(1, 1, -1, 1, 1, -1, -1, -1, -1, 1, -1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, 1, 1, -1)
r4 = c(1, 1, -1, 1, -1, 1, -1, -1, -1, 1, 1, -1, 1, -1, 1, 1, -1, 1, 1, 1, -1, -1, -1, -1)
r5 = c(-1, -1, 1, 1, 1, 1, -1, 1, 1, -1, -1, -1, 1, -1, 1, 1, 1, -1, -1, 1, -1, 1, 1, 1)
r6 = c(-1, -1, 1, 1, 1, 1, 1, -1, 1, 1, 1, -1, -1, 1, 1, -1, 1, 1, 1, 1, 1, 1, -1, -1)
r7 = c(-1, -1, -1, -1, 1, -1, -1, 1, -1, 1, -1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, -1, -1, 1)
r8 = c(-1, 1, 1, -1, -1, 1, -1, 1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, 1, -1, 1, 1, 1, -1)
r9 = c(-1, -1, -1, -1, -1, 1, 1, -1, -1, -1, 1, 1, -1, -1, 1, -1, 1, 1, -1, -1, -1, -1, 1, 1)
r10 = c(1, 1, 1, 1, -1, 1, 1, 1, -1, -1, -1, 1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, -1, 1)
r11 = c(-1, 1, -1, 1, 1, -1, -1, 1, 1, -1, 1, -1, -1, 1, -1, -1, -1, 1, 1, -1, -1, -1, 1, 1)
r12 = c(1, -1, -1, -1, 1, 1, 1, -1, 1, 1, 1, 1, 1, -1, -1, 1, -1, -1, 1, -1, 1, 1, 1, 1)
r13 = c(1, 1, 1, 1, 1, -1, 1, -1, 1, -1, -1, 1, -1, -1, -1, -1, -1, 1, -1, 1, 1, -1, 1, -1)
r14 = c(-1, -1, 1, -1, -1, -1, -1, -1, -1, -1, 1, 1, -1, 1, -1, -1, -1, -1, -1, 1, -1, 1, -1, -1)

D = rbind(r1, r2, r3, r4, r5, r6, r7, r8, r9, r10, r11, r12, r13, r14)
X = D[,-16]

n=nrow(X)
p=ncol(X)
```

```
nwant=1000
nrow=2.5*nwant

mynsis=n-2 #Model 1 only
#mynsis=n-1 #Models 2 and 3 only

AIC.SIS = matrix(rep(0), nrow, p)
AIC.ISIS = matrix(rep(0), nrow, p)
AIC.SISSCAD = matrix(rep(0), nrow, p)
AIC.ISISSCAD = matrix(rep(0), nrow, p)

BIC.SIS = matrix(rep(0), nrow, p)
BIC.ISIS = matrix(rep(0), nrow, p)
BIC.SISSCAD = matrix(rep(0), nrow, p)
BIC.ISISSCAD = matrix(rep(0), nrow, p)

AIC.SISSCAD.P = matrix(rep(0), nrow, p+1)
AIC.ISISSCAD.P = matrix(rep(0), nrow, p+1)
BIC.SISSCAD.P = matrix(rep(0), nrow, p+1)
BIC.ISISSCAD.P = matrix(rep(0), nrow, p+1)

#Model 1
b = c(8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
sigcol = c(1,12)
numcol = length(sigcol)
nonsigcol = c(2:11,13:23)

#Model 2
#b = c(10, 9, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
#sigcol = c(1,2,3)
#numcol = length(sigcol)
#nonsigcol = c(4:23)

#Model 3
#b = c(-20, 0, 12, 0, 10, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0)
#sigcol = c(1,3,5,7,16)
#numcol = length(sigcol)
#nonsigcol = c(2,4,6,8:15,17:23)

set.seed(0)
```

```
#AIC and BIC Tuning Methods
for(i in 1:nrow) {
AIC.SIS[i,]=0
AIC.ISIS[i,]=0
AIC.SISSCAD[i,]=0
AIC.ISISSCAD[i,]=0
BIC.SIS[i,]=0
BIC.ISIS[i,]=0
BIC.SISSCAD[i,]=0
BIC.ISISSCAD[i,]=0
AIC.SISSCAD.P[i,]=0
AIC.ISISSCAD.P[i,]=0
BIC.SISSCAD.P[i,]=0
BIC.ISISSCAD.P[i,]=0

error = matrix(rnorm(n, mean=0, sd=1), n, 1)
y = X%*%b + error

AIC=try(SIS(data=list(X, y), model='glm', vartype=0, nsis=mynsis, family=gaussian(),
rank.method='coeff', eps0=1e-5, tune.method='AIC', post.tune.method='AIC', DOISIS=TRUE),
TRUE)

if (length(AIC)==1) {
AIC.SIS[i,]=NaN
AIC.ISIS[i,]=NaN
AIC.SISSCAD[i,]=NaN
AIC.ISISSCAD[i,]=NaN
AIC.SISSCAD.P[i,]=NaN
AIC.ISISSCAD.P[i,]=NaN }
else {
AIC.SIS[i,AIC$SISind]=1
AIC.ISIS[i,AIC$ISISind]=1
AIC.SISSCAD.P[i,] = AIC$SIScoef
AIC.SISest = AIC$SIScoef[-1]
AIC.SISparam = abs(AIC.SISest) > 0
for(j in 1:p) {
if (AIC.SISparam[j]==TRUE) {
AIC.SISSCAD[i,j]=1
}
else {
AIC.SISSCAD[i,j]=0
}
```

```
}
AIC.ISISSCAD.P[i,] = AIC$ISIScoef
AIC.ISISest = AIC$ISIScoef[-1]
AIC.ISISparam = abs(AIC.ISISest) > 0
for(j in 1:p) {
if (AIC.ISISparam[j]==TRUE) {
AIC.ISISSCAD[i,j]=1
}
else {
AIC.ISISSCAD[i,j]=0
}
}
}
```

```
BIC=try(SIS(data=list(X, y), model='glm', vartype=0, nsis=mynsis, family=gaussian(),
rank.method='coeff', eps0=1e-5, tune.method='BIC', post.tune.method='BIC', DOISIS=TRUE),
TRUE)

if (length(BIC)==1) {
BIC.SIS[i,]=NaN
BIC.ISIS[i,]=NaN
BIC.SISSCAD[i,]=NaN
BIC.ISISSCAD[i,]=NaN
BIC.SISSCAD.P[i,]=NaN
BIC.ISISSCAD.P[i,]=NaN }
else {
BIC.SIS[i,BIC$SISind]=1
BIC.ISIS[i,BIC$ISISind]=1
BIC.SISSCAD.P[i,] = BIC$SIScoef
BIC.SISest = BIC$SIScoef[-1]
BIC.SISparam = abs(BIC.SISest) > 0
for(j in 1:p) {
if (BIC.SISparam[j]==TRUE) {
BIC.SISSCAD[i,j]=1
}
else {
BIC.SISSCAD[i,j]=0
}
}
BIC.ISISSCAD.P[i,] = BIC$ISIScoef
BIC.ISISest = BIC$ISIScoef[-1]
BIC.ISISparam = abs(BIC.ISISest) > 0
```

```
for(j in 1:p) {
if (BIC.ISISparam[j]==TRUE) {
BIC.ISISSCAD[i,j]=1
}
else {
BIC.ISISSCAD[i,j]=0
}
}
}
}

#One-step SIS important factors
AIC.SIS
BIC.SIS

#ISIS important factors
AIC.ISIS
BIC.ISIS

#SIS-SCAD important factors
AIC.SISSCAD
BIC.SISSCAD

#ISIS-SCAD important factors
AIC.ISISSCAD
BIC.ISISSCAD

#Getting rid of the NaN rows
AIC.SIS2 = AIC.SIS[AIC.SIS[,1]!="NaN",]
BIC.SIS2 = BIC.SIS[BIC.SIS[,1]!="NaN",]
AIC.ISIS2 = AIC.ISIS[AIC.ISIS[,1]!="NaN",]
BIC.ISIS2 = BIC.ISIS[BIC.ISIS[,1]!="NaN",]
AIC.SISSCAD2 = AIC.SISSCAD[AIC.SISSCAD[,1]!="NaN",]
BIC.SISSCAD2 = BIC.SISSCAD[BIC.SISSCAD[,1]!="NaN",]
AIC.ISISSCAD2 = AIC.ISISSCAD[AIC.ISISSCAD[,1]!="NaN",]
BIC.ISISSCAD2 = BIC.ISISSCAD[BIC.ISISSCAD[,1]!="NaN",]
AIC.SISSCAD.P2 = AIC.SISSCAD.P[AIC.SISSCAD.P[,1]!="NaN",]
BIC.SISSCAD.P2 = BIC.SISSCAD.P[BIC.SISSCAD.P[,1]!="NaN",]
AIC.ISISSCAD.P2 = AIC.ISISSCAD.P[AIC.ISISSCAD.P[,1]!="NaN",]
BIC.ISISSCAD.P2 = BIC.ISISSCAD.P[BIC.ISISSCAD.P[,1]!="NaN",]
```

```
#Pulling the first "nwant" successful instances
AIC.SIS = AIC.SIS2[1:nwant,]
BIC.SIS = BIC.SIS2[1:nwant,]
AIC.ISIS = AIC.ISIS2[1:nwant,]
BIC.ISIS = BIC.ISIS2[1:nwant,]
AIC.SISSCAD = AIC.SISSCAD2[1:nwant,]
BIC.SISSCAD = BIC.SISSCAD2[1:nwant,]
AIC.ISISSCAD = AIC.ISISSCAD2[1:nwant,]
BIC.ISISSCAD = BIC.ISISSCAD2[1:nwant,]
AIC.SISSCAD.P = AIC.SISSCAD.P2[1:nwant,]
BIC.SISSCAD.P = BIC.SISSCAD.P2[1:nwant,]
AIC.ISISSCAD.P = AIC.ISISSCAD.P2[1:nwant,]
BIC.ISISSCAD.P = BIC.ISISSCAD.P2[1:nwant,]

#One-step SIS important factors
AIC.SIS
BIC.SIS

#ISIS important factors
AIC.ISIS
BIC.ISIS

#SIS-SCAD important factors
AIC.SISSCAD
BIC.SISSCAD

#ISIS-SCAD important factors
AIC.ISISSCAD
BIC.ISISSCAD

AIC.SIS.IR = matrix(rep(0), nwant, 4)
BIC.SIS.IR = matrix(rep(0), nwant, 4)
AIC.ISIS.IR = matrix(rep(0), nwant, 4)
BIC.ISIS.IR = matrix(rep(0), nwant, 4)
AIC.SISSCAD.IR = matrix(rep(0), nwant, 4)
BIC.SISSCAD.IR = matrix(rep(0), nwant, 4)
AIC.ISISSCAD.IR = matrix(rep(0), nwant, 4)
BIC.ISISSCAD.IR = matrix(rep(0), nwant, 4)
```

```
for (i in 1:nwant) {
if (sum(AIC.SIS[i,sigcol]) == numcol) {
AIC.SIS.IR[i,1]=1 }
else { AIC.SIS.IR[i,1]=0 }
if (sum(AIC.SIS[i,nonsigcol]) == 0) {
AIC.SIS.IR[i,2]=1 }
else { AIC.SIS.IR[i,2]=0 }
if (sum(AIC.SIS.IR[i,c(1:2)]) == 2) {
AIC.SIS.IR[i,3]=1 }
else { AIC.SIS.IR[i,3]=0 }
AIC.SIS.IR[i,4]=sum(AIC.SIS[i,])
}

for (i in 1:nwant) {
if (sum(BIC.SIS[i,sigcol]) == numcol) {
BIC.SIS.IR[i,1]=1 }
else { BIC.SIS.IR[i,1]=0 }
if (sum(BIC.SIS[i,nonsigcol]) == 0) {
BIC.SIS.IR[i,2]=1 }
else { BIC.SIS.IR[i,2]=0 }
if (sum(BIC.SIS.IR[i,c(1:2)]) == 2) {
BIC.SIS.IR[i,3]=1 }
else { BIC.SIS.IR[i,3]=0 }
BIC.SIS.IR[i,4]=sum(BIC.SIS[i,])
}

for (i in 1:nwant) {
if (sum(AIC.ISIS[i,sigcol]) == numcol) {
AIC.ISIS.IR[i,1]=1 }
else { AIC.ISIS.IR[i,1]=0 }
if (sum(AIC.ISIS[i,nonsigcol]) == 0) {
AIC.ISIS.IR[i,2]=1 }
else { AIC.ISIS.IR[i,2]=0 }
if (sum(AIC.ISIS.IR[i,c(1:2)]) == 2) {
AIC.ISIS.IR[i,3]=1 }
else { AIC.ISIS.IR[i,3]=0 }
AIC.ISIS.IR[i,4]=sum(AIC.ISIS[i,])
}
```

```
for (i in 1:nwant) {
if (sum(BIC.ISIS[i,sigcol]) == numcol) {
BIC.ISIS.IR[i,1]=1 }
else { BIC.ISIS.IR[i,1]=0 }
if (sum(BIC.ISIS[i,nonsigcol]) == 0) {
BIC.ISIS.IR[i,2]=1 }
else { BIC.ISIS.IR[i,2]=0 }
if (sum(BIC.ISIS.IR[i,c(1:2)]) == 2) {
BIC.ISIS.IR[i,3]=1 }
else { BIC.ISIS.IR[i,3]=0 }
BIC.ISIS.IR[i,4]=sum(BIC.ISIS[i,])
}

for (i in 1:nwant) {
if (sum(AIC.SISSCAD[i,sigcol]) == numcol) {
AIC.SISSCAD.IR[i,1]=1 }
else { AIC.SISSCAD.IR[i,1]=0 }
if (sum(AIC.SISSCAD[i,nonsigcol]) == 0) {
AIC.SISSCAD.IR[i,2]=1 }
else { AIC.SISSCAD.IR[i,2]=0 }
if (sum(AIC.SISSCAD.IR[i,c(1:2)]) == 2) {
AIC.SISSCAD.IR[i,3]=1 }
else { AIC.SISSCAD.IR[i,3]=0 }
AIC.SISSCAD.IR[i,4]=sum(AIC.SISSCAD[i,])
}

for (i in 1:nwant) {
if (sum(BIC.SISSCAD[i,sigcol]) == numcol) {
BIC.SISSCAD.IR[i,1]=1 }
else { BIC.SISSCAD.IR[i,1]=0 }
if (sum(BIC.SISSCAD[i,nonsigcol]) == 0) {
BIC.SISSCAD.IR[i,2]=1 }
else { BIC.SISSCAD.IR[i,2]=0 }
if (sum(BIC.SISSCAD.IR[i,c(1:2)]) == 2) {
BIC.SISSCAD.IR[i,3]=1 }
else { BIC.SISSCAD.IR[i,3]=0 }
BIC.SISSCAD.IR[i,4]=sum(BIC.SISSCAD[i,])
}
```

```
for (i in 1:nwant) {
if (sum(AIC.ISISSCAD[i,sigcol]) == numcol) {
AIC.ISISSCAD.IR[i,1]=1 }
else { AIC.ISISSCAD.IR[i,1]=0 }
if (sum(AIC.ISISSCAD[i,nonsigcol]) == 0) {
AIC.ISISSCAD.IR[i,2]=1 }
else { AIC.ISISSCAD.IR[i,2]=0 }
if (sum(AIC.ISISSCAD.IR[i,c(1:2)]) == 2) {
AIC.ISISSCAD.IR[i,3]=1 }
else { AIC.ISISSCAD.IR[i,3]=0 }
AIC.ISISSCAD.IR[i,4]=sum(AIC.ISISSCAD[i,])
}

for (i in 1:nwant) {
if (sum(BIC.ISISSCAD[i,sigcol]) == numcol) {
BIC.ISISSCAD.IR[i,1]=1 }
else { BIC.ISISSCAD.IR[i,1]=0 }
if (sum(BIC.ISISSCAD[i,nonsigcol]) == 0) {
BIC.ISISSCAD.IR[i,2]=1 }
else { BIC.ISISSCAD.IR[i,2]=0 }
if (sum(BIC.ISISSCAD.IR[i,c(1:2)]) == 2) {
BIC.ISISSCAD.IR[i,3]=1 }
else { BIC.ISISSCAD.IR[i,3]=0 }
BIC.ISISSCAD.IR[i,4]=sum(BIC.ISISSCAD[i,])
}

#IR = Identification Rate = % of simulations identifying only the active factors
(sum(AIC.SIS.IR[,3])/nwant)*100
(sum(BIC.SIS.IR[,3])/nwant)*100
(sum(AIC.ISIS.IR[,3])/nwant)*100
(sum(BIC.ISIS.IR[,3])/nwant)*100
(sum(AIC.SISSCAD.IR[,3])/nwant)*100
(sum(BIC.SISSCAD.IR[,3])/nwant)*100
(sum(AIC.ISISSCAD.IR[,3])/nwant)*100
(sum(BIC.ISISSCAD.IR[,3])/nwant)*100
```

```
#C = Coverage = % of simulations that identified all the active factors plus maybe others
(sum(AIC.SIS.IR[,1])/nwant)*100
(sum(BIC.SIS.IR[,1])/nwant)*100
(sum(AIC.ISIS.IR[,1])/nwant)*100
(sum(BIC.ISIS.IR[,1])/nwant)*100
(sum(AIC.SISSCAD.IR[,1])/nwant)*100
(sum(BIC.SISSCAD.IR[,1])/nwant)*100
(sum(AIC.ISISSCAD.IR[,1])/nwant)*100
(sum(BIC.ISISSCAD.IR[,1])/nwant)*100

#P = Power = Average % of active factors correctly declared as active
(sum(AIC.SIS[,sigcol])/(numcol*nwant))*100
(sum(BIC.SIS[,sigcol])/(numcol*nwant))*100
(sum(AIC.ISIS[,sigcol])/(numcol*nwant))*100
(sum(BIC.ISIS[,sigcol])/(numcol*nwant))*100
(sum(AIC.SISSCAD[,sigcol])/(numcol*nwant))*100
(sum(BIC.SISSCAD[,sigcol])/(numcol*nwant))*100
(sum(AIC.ISISSCAD[,sigcol])/(numcol*nwant))*100
(sum(BIC.ISISSCAD[,sigcol])/(numcol*nwant))*100

#T1 = Type 1 error rate = Average % of inactive factors incorrectly declared as active
(sum(AIC.SIS[,nonsigcol])/(length(nonsigcol)*nwant))*100
(sum(BIC.SIS[,nonsigcol])/(length(nonsigcol)*nwant))*100
(sum(AIC.ISIS[,nonsigcol])/(length(nonsigcol)*nwant))*100
(sum(BIC.ISIS[,nonsigcol])/(length(nonsigcol)*nwant))*100
(sum(AIC.SISSCAD[,nonsigcol])/(length(nonsigcol)*nwant))*100
(sum(BIC.SISSCAD[,nonsigcol])/(length(nonsigcol)*nwant))*100
(sum(AIC.ISISSCAD[,nonsigcol])/(length(nonsigcol)*nwant))*100
(sum(BIC.ISISSCAD[,nonsigcol])/(length(nonsigcol)*nwant))*100

#M = Mean # factors declared as active
sum(AIC.SIS.IR[,4])/nwant
sum(BIC.SIS.IR[,4])/nwant
sum(AIC.ISIS.IR[,4])/nwant
sum(BIC.ISIS.IR[,4])/nwant
sum(AIC.SISSCAD.IR[,4])/nwant
sum(BIC.SISSCAD.IR[,4])/nwant
sum(AIC.ISISSCAD.IR[,4])/nwant
sum(BIC.ISISSCAD.IR[,4])/nwant
```

```
#V = Variance of # factors declared as active
var(AIC.SIS.IR[,4])
var(BIC.SIS.IR[,4])
var(AIC.ISIS.IR[,4])
var(BIC.ISIS.IR[,4])
var(AIC.SISSCAD.IR[,4])
var(BIC.SISSCAD.IR[,4])
var(AIC.ISISSCAD.IR[,4])
var(BIC.ISISSCAD.IR[,4])

#Probability that a factor was identified as active
M = matrix(rep(0), p, 8)
for(i in 1:p) {
M[i,1] = sum(AIC.SIS[,i])/nwant*100
M[i,2] = sum(BIC.SIS[,i])/nwant*100
M[i,3] = sum(AIC.SISSCAD[,i])/nwant*100
M[i,4] = sum(BIC.SISSCAD[,i])/nwant*100
M[i,5] = sum(AIC.ISIS[,i])/nwant*100
M[i,6] = sum(BIC.ISIS[,i])/nwant*100
M[i,7] = sum(AIC.ISISSCAD[,i])/nwant*100
M[i,8] = sum(BIC.ISISSCAD[,i])/nwant*100 }

#Average parameter estimates
P = matrix(rep(0), p+1, 4)
for(i in 1:(p+1)) {
P[i,1] = mean(AIC.SISSCAD.P[,i])
P[i,2] = mean(BIC.SISSCAD.P[,i])
P[i,3] = mean(AIC.ISISSCAD.P[,i])
P[i,4] = mean(BIC.ISISSCAD.P[,i]) }

M
P
```

Appendix B

R Code - Williams Rubber Data with Original Response

```
#Load package 'SIS' first
library(SIS)

#Half-fraction of Plackett Burman design from Lin (1993)
r1 = c(1, 1, 1, -1, -1, -1, 1, 1, 1, 1, 1, -1, 1, -1, -1, 1, 1, -1, -1, 1, -1, -1, -1, 1)
r2 = c(1, -1, -1, -1, -1, -1, 1, 1, 1, -1, -1, -1, 1, 1, 1, 1, -1, 1, -1, -1, 1, 1, -1, -1)
r3 = c(1, 1, -1, 1, 1, -1, -1, -1, -1, 1, -1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, 1, 1, -1)
r4 = c(1, 1, -1, 1, -1, 1, -1, -1, -1, 1, 1, -1, 1, -1, 1, 1, -1, 1, 1, 1, -1, -1, -1, -1)
r5 = c(-1, -1, 1, 1, 1, 1, -1, 1, 1, -1, -1, -1, 1, -1, 1, 1, 1, -1, -1, 1, -1, 1, 1, 1)
r6 = c(-1, -1, 1, 1, 1, 1, 1, -1, 1, 1, 1, -1, -1, 1, 1, -1, 1, 1, 1, 1, 1, 1, -1, -1)
r7 = c(-1, -1, -1, -1, 1, -1, -1, 1, -1, 1, -1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, -1, -1, 1)
r8 = c(-1, 1, 1, -1, -1, 1, -1, 1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, 1, -1, 1, 1, 1, -1)
r9 = c(-1, -1, -1, -1, -1, 1, 1, -1, -1, -1, 1, 1, -1, -1, 1, -1, 1, 1, -1, -1, -1, -1, 1, 1)
r10 = c(1, 1, 1, 1, -1, 1, 1, 1, -1, -1, -1, 1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, -1, 1)
r11 = c(-1, 1, -1, 1, 1, -1, -1, 1, 1, -1, 1, -1, -1, 1, -1, -1, -1, 1, 1, -1, -1, -1, 1, 1)
r12 = c(1, -1, -1, -1, 1, 1, 1, -1, 1, 1, 1, 1, 1, -1, -1, 1, -1, -1, 1, -1, 1, 1, 1, 1)
r13 = c(1, 1, 1, 1, 1, -1, 1, -1, 1, -1, -1, 1, -1, -1, -1, -1, -1, 1, -1, 1, 1, -1, 1, -1)
r14 = c(-1, -1, 1, -1, -1, -1, -1, -1, -1, -1, 1, 1, -1, 1, -1, -1, -1, -1, -1, 1, -1, 1, -1, -1)

D = rbind(r1, r2, r3, r4, r5, r6, r7, r8, r9, r10, r11, r12, r13, r14)
y = rbind(133, 62, 45, 52, 56, 47, 88, 193, 32, 53, 276, 145, 130, 127)

X = D[,-16]
```

```
n = nrow(X)
p = ncol(X)

#Change this for different max factors
mynsis = n-1

BIC = SIS(data=list(X, y), model='glm', vartype=0, nsis=mynsis, family=gaussian(),
rank.method='coeff', eps0=1e-5, tune.method='BIC', post.tune.method='BIC',
DOISIS=TRUE)

BIC.SIS = matrix(rep(0), 1, p)
BIC.ISIS = matrix(rep(0), 1, p)
BIC.SISSCAD = matrix(rep(0), 1, p)
BIC.ISISSCAD = matrix(rep(0), 1, p)

BIC.SIS[BIC$SISind]=1
BIC.ISIS[BIC$ISISind]=1

BIC.SISest = BIC$SIScoef[-1]
BIC.SISparam = abs(BIC.SISest) ¿ 0
for(j in 1:p) {
if (BIC.SISparam[j]==TRUE) {
BIC.SISSCAD[,j]=1
}
else {
BIC.SISSCAD[,j]=0
}
}

BIC.ISISest = BIC$ISIScoef[-1]
BIC.ISISparam = abs(BIC.ISISest) ¿ 0
for(j in 1:p) {
if (BIC.ISISparam[j]==TRUE) {
BIC.ISISSCAD[,j]=1
}
else {
BIC.ISISSCAD[,j]=0
}
}
```

```
BIC.SIS
BIC.ISIS
BIC.SISSCAD
BIC.ISISSCAD

BIC$SIScoef
BIC$ISIScoef

#Remember - Factors 16 - 23 are really factors 17 - 24.
#Also, factor 13 is indistinguishable from factor 16.
#Need to convert back to original numbers after running this analysis.

SStot = sum((y - mean(y))*(y - mean(y)))

yhat.SIS = BIC$SIScoef[1] + X%*%BIC$SIScoef[-1]
SSerr.SIS = sum((y - yhat.SIS)*(y - yhat.SIS))
R2.SIS = 1 - (SSerr.SIS/SStot)
R2.SIS

ADJ.R2.SIS = 1 - ((1-R2.SIS)*(n-1)/(n-sum(BIC.SISSCAD)-1))
ADJ.R2.SIS

yhat.ISIS = BIC$ISIScoef[1] + X%*%BIC$ISIScoef[-1]
SSerr.ISIS = sum((y - yhat.ISIS)*(y - yhat.ISIS))
R2.ISIS = 1 - (SSerr.ISIS/SStot)
R2.ISIS

ADJ.R2.ISIS = 1 - ((1-R2.ISIS)*(n-1)/(n-sum(BIC.ISISSCAD)-1))
ADJ.R2.ISIS
```

Appendix C

R Code - Marley and Woods Designs

```
#Load package 'SIS' first
library(SIS)

#Bayesian D-optimal, 22 factors, 18 runs
#r1 = c(-1, 1, -1, -1, -1, -1, -1, 1, 1, -1, 1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1)
#r2 = c(1, 1, -1, -1, -1, -1, 1, 1, -1, -1, -1, 1, 1, 1, 1, 1, -1, -1, 1, 1, 1, 1)
#r3 = c(1, 1, -1, 1, -1, 1, -1, -1, 1, 1, 1, -1, -1, 1, -1, 1, -1, -1, 1, -1, -1, 1)
#r4 = c(-1, -1, -1, 1, -1, -1, -1, -1, -1, 1, 1, 1, 1, 1, 1, -1, -1, 1, 1, 1, -1, -1)
#r5 = c(1, -1, -1, -1, 1, 1, -1, -1, -1, -1, 1, 1, 1, 1, -1, -1, 1, -1, -1, -1, 1, 1)
#r6 = c(-1, 1, 1, 1, 1, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, -1, 1)
#r7 = c(-1, -1, -1, -1, 1, -1, -1, 1, 1, 1, 1, 1, -1, -1, 1, 1, -1, -1, -1, -1, 1, -1)
#r8 = c(-1, -1, 1, -1, -1, 1, 1, 1, 1, -1, -1, -1, 1, 1, 1, -1, -1, -1, -1, -1, -1, -1)
#r9 = c(1, 1, 1, -1, -1, 1, -1, 1, -1, 1, -1, 1, -1, -1, -1, -1, 1, 1, 1, -1, 1, -1)
#r10 = c(-1, -1, -1, 1, -1, 1, 1, -1, -1, 1, -1, -1, -1, -1, 1, -1, 1, -1, -1, 1, 1, 1)
#r11 = c(1, -1, 1, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, 1, 1, 1, -1, -1, -1, 1)
#r12 = c(1, 1, -1, -1, 1, 1, -1, -1, -1, -1, -1, -1, -1, -1, 1, 1, -1, 1, -1, 1, -1, -1)
#r13 = c(-1, 1, -1, 1, 1, 1, 1, 1, -1, -1, 1, -1, 1, -1, -1, 1, 1, 1, 1, -1, 1, -1)
#r14 = c(1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, -1, -1, -1, -1, -1, -1, 1, 1, 1, -1)
#r15 = c(-1, -1, 1, 1, -1, 1, -1, -1, 1, -1, -1, 1, 1, -1, -1, 1, -1, 1, -1, 1, 1, 1)
#r16 = c(1, -1, 1, 1, 1, -1, -1, 1, -1, 1, -1, -1, 1, 1, -1, 1, 1, -1, -1, 1, -1, -1)
#r17 = c(-1, -1, -1, -1, 1, 1, 1, -1, 1, 1, -1, 1, -1, 1, -1, 1, 1, 1, 1, 1, -1, 1)
#r18 = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, -1, -1, 1, -1, 1, 1, 1)
#X = rbind(r1, r2, r3, r4, r5, r6, r7, r8, r9, r10, r11, r12, r13, r14, r15, r16, r17, r18)
```

```
#Bayesian D-optimal, 24 factors, 14 runs
#r1 = c(1, 1, 1, -1, -1, -1, -1, -1, 1, -1, -1, -1, -1, 1, -1, 1, -1, 1, 1, 1, -1, -1, -1, 1)
#r2 = c(-1, 1, -1, 1, 1, 1, 1, 1, 1, 1, -1, -1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, -1, 1)
#r3 = c(1, 1, -1, -1, 1, 1, 1, 1, 1, -1, 1, -1, -1, -1, -1, -1, 1, 1, -1, -1, -1, 1, 1, -1)
#r4 = c(1, 1, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, 1, -1, 1, -1, -1, -1, -1, 1, 1, 1, 1, 1)
#r5 = c(-1, -1, 1, 1, 1, -1, -1, -1, 1, -1, -1, 1, 1, -1, -1, -1, 1, 1, -1, -1, 1, 1, -1, 1)
#r6 = c(1, -1, -1, -1, 1, -1, -1, 1, -1, -1, -1, 1, 1, 1, 1, 1, 1, -1, -1, 1, -1, 1, -1, -1)
#r7 = c(1, -1, 1, 1, -1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1)
#r8 = c(-1, -1, -1, -1, -1, 1, 1, -1, -1, 1, -1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, 1, -1, -1)
#r9 = c(1, -1, 1, 1, -1, -1, 1, 1, -1, 1, 1, -1, -1, -1, 1, -1, -1, -1, -1, 1, -1, 1, -1, 1)
#r10 = c(-1, -1, -1, -1, -1, -1, 1, -1, -1, 1, 1, 1, 1, -1, -1, 1, -1, 1, -1, -1, -1, -1, 1, -1)
#r11 = c(-1, 1, 1, -1, 1, -1, -1, 1, -1, 1, 1, -1, 1, 1, -1, -1, 1, -1, 1, 1, 1, -1, 1, 1)
#r12 = c(1, -1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, 1, -1, 1, -1, -1, -1, 1, -1, -1, -1, 1, -1)
#r13 = c(-1, 1, -1, 1, -1, 1, -1, 1, -1, -1, 1, 1, -1, -1, -1, 1, -1, -1, 1, -1, 1, 1, -1, -1)
#r14 = c(-1, 1, 1, 1, 1, -1, 1, -1, 1, 1, -1, 1, -1, -1, 1, 1, 1, -1, 1, 1, -1, 1, 1, -1)
#X = rbind(r1, r2, r3, r4, r5, r6, r7, r8, r9, r10, r11, r12, r13, r14)

#Bayesian D-optimal, 26 factors, 12 runs
r1 = c(-1, -1, -1, 1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1, 1, 1, 1, -1, -1, -1, -1)
r2 = c(1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, -1, 1, -1, 1, 1, 1, 1, -1, 1, 1, 1, 1, -1, -1, -1)
r3 = c(-1, -1, 1, -1, 1, 1, 1, -1, 1, -1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1, -1, 1, -1, 1, 1)
r4 = c(-1, 1, 1, 1, -1, 1, -1, 1, -1, 1, 1, -1, 1, -1, 1, -1, -1, 1, 1, -1, -1, -1, -1, -1, 1, 1)
r5 = c(1, -1, 1, -1, -1, 1, -1, 1, 1, -1, -1, -1, 1, 1, -1, -1, -1, 1, -1, 1, 1, -1, 1, 1, 1, -1)
r6 = c(1, -1, -1, -1, 1, 1, 1, 1, -1, 1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1, 1, -1, -1, 1, -1, 1)
r7 = c(-1, 1, -1, -1, 1, -1, -1, -1, -1, 1, -1, -1, -1, 1, 1, 1, -1, 1, 1, 1, -1, 1, 1, 1, 1, -1)
r8 = c(1, 1, -1, 1, -1, 1, -1, -1, 1, -1, 1, 1, -1, -1, -1, 1, 1, 1, 1, -1, 1, 1, -1, 1, 1, -1)
r9 = c(-1, 1, -1, -1, 1, -1, -1, -1, 1, 1, -1, -1, 1, -1, -1, -1, 1, -1, -1, -1, 1, -1, -1, 1, -1, 1)
r10 = c(1, -1, 1, 1, -1, -1, -1, 1, -1, 1, -1, 1, 1, 1, -1, 1, 1, -1, 1, 1, -1, 1, -1, 1, -1, 1)
r11 = c(-1, 1, 1, -1, -1, -1, 1, -1, -1, -1, 1, 1, -1, 1, -1, -1, 1, 1, -1, 1, -1, -1, -1, -1, -1, -1)
r12 = c(1, -1, -1, 1, -1, -1, -1, -1, 1, -1, 1, -1, -1, 1, 1, -1, -1, -1, -1, -1, -1, 1, 1, -1, -1, 1)
X = rbind(r1, r2, r3, r4, r5, r6, r7, r8, r9, r10, r11, r12)
```

#$E(s^2)$-optimal, 22 factors, 18 runs
#r1 = c(-1, -1, -1, -1, 1, -1, -1, -1, 1, -1, -1, 1, -1, -1, 1, 1, -1, -1, -1, 1, 1, 1)
#r2 = c(1, 1, 1, 1, -1, -1, -1, 1, 1, -1, 1, 1, -1, -1, 1, -1, 1, -1, 1, 1, -1, 1)
#r3 = c(-1, -1, 1, -1, -1, 1, 1, 1, 1, -1, -1, 1, -1, 1, -1, 1, 1, 1, 1, -1, 1, -1)
#r4 = c(1, 1, -1, 1, 1, -1, -1, 1, 1, 1, 1, 1, -1, 1, -1, -1, -1, 1, -1, -1, 1, -1)
#r5 = c(1, 1, 1, 1, -1, -1, 1, -1, -1, 1, -1, 1, 1, -1, -1, 1, 1, 1, -1, 1, 1, -1)
#r6 = c(1, -1, -1, 1, 1, 1, 1, 1, -1, -1, 1, -1, -1, -1, -1, 1, 1, -1, -1, -1, 1, 1)
#r7 = c(-1, 1, -1, -1, -1, 1, -1, -1, 1, -1, 1, -1, 1, -1, -1, -1, 1, 1, -1, -1, -1, 1)
#r8 = c(-1, -1, -1, 1, 1, -1, 1, -1, -1, -1, -1, -1, -1, 1, 1, -1, 1, 1, -1, 1, -1, -1)
#r9 = c(1, -1, -1, -1, -1, -1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, -1, 1, -1, 1, -1, 1)
#r10 = c(1, 1, 1, 1, 1, -1, 1, -1, 1, -1, -1, -1, 1, 1, -1, 1, -1, -1, 1, -1, -1, 1)
#r11 = c(1, -1, 1, -1, 1, 1, -1, 1, -1, 1, -1, 1, 1, 1, -1, -1, 1, -1, -1, 1, -1, 1)
#r12 = c(1, 1, -1, -1, 1, 1, 1, 1, -1, -1, -1, 1, 1, -1, 1, -1, -1, 1, 1, -1, -1, -1)
#r13 = c(-1, -1, 1, 1, -1, 1, -1, -1, -1, -1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1)
#r14 = c(1, -1, 1, 1, -1, 1, -1, -1, -1, 1, -1, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, 1)
#r15 = c(-1, 1, 1, -1, 1, -1, -1, 1, -1, 1, 1, -1, -1, -1, -1, 1, -1, 1, 1, 1, -1, -1)
#r16 = c(-1, 1, -1, -1, -1, 1, 1, -1, -1, 1, 1, 1, -1, 1, -1, -1, -1, -1, 1, 1, 1, 1)
#r17 = c(-1, -1, 1, -1, 1, -1, 1, -1, 1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1)
#r18 = c(-1, 1, -1, 1, -1, 1, -1, 1, 1, 1, -1, -1, 1, 1, 1, 1, 1, -1, 1, 1, 1, -1)
#X = rbind(r1, r2, r3, r4, r5, r6, r7, r8, r9, r10, r11, r12, r13, r14, r15, r16, r17, r18)

#$E(s^2)$-optimal, 24 factors, 14 runs
#r1 = c(1, 1, -1, 1, -1, 1, -1, 1, 1, -1, -1, 1, -1, 1, -1, -1, -1, 1, -1, -1, -1, -1, 1, 1)
#r2 = c(-1, -1, 1, 1, -1, -1, 1, -1, 1, 1, -1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1)
#r3 = c(-1, 1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, -1, 1, -1, 1, 1, 1)
#r4 = c(1, -1, -1, -1, 1, 1, 1, 1, 1, -1, -1, 1, 1, 1, -1, 1, 1, 1, 1, 1, -1, 1, -1, -1)
#r5 = c(1, 1, -1, -1, 1, -1, -1, -1, -1, 1, -1, -1, 1, 1, 1, 1, 1, 1, -1, -1, 1, -1, 1, -1)
#r6 = c(-1, -1, 1, -1, -1, -1, -1, -1, -1, -1, 1, 1, -1, 1, 1, -1, 1, -1, 1, -1, -1, 1, 1, -1)
#r7 = c(-1, -1, 1, 1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1, 1, -1, 1, -1, -1, 1, 1, -1, -1, 1)
#r8 = c(1, -1, -1, -1, -1, 1, 1, 1, 1, 1, 1, -1, -1, -1, 1, 1, 1, -1, 1, -1, 1, -1, 1, 1)
#r9 = c(-1, -1, -1, -1, -1, 1, -1, -1, -1, -1, -1, -1, 1, -1, -1, 1, -1, -1, -1, -1, -1, 1, -1, 1)
#r10 = c(1, 1, 1, -1, -1, -1, -1, 1, 1, 1, 1, -1, 1, 1, -1, -1, -1, -1, -1, 1, 1, 1, -1, -1)
#r11 = c(-1, -1, -1, 1, 1, -1, -1, -1, 1, -1, 1, -1, -1, -1, -1, -1, -1, 1, 1, 1, 1, -1, -1, -1)
#r12 = c(-1, 1, 1, 1, -1, -1, 1, 1, -1, 1, -1, -1, -1, -1, -1, 1, 1, 1, 1, -1, -1, -1, -1, -1)
#r13 = c(1, 1, 1, -1, 1, 1, 1, -1, -1, -1, 1, 1, 1, -1, 1, -1, -1, 1, 1, -1, 1, -1, -1, 1)
#r14 = c(1, 1, 1, 1, 1, 1, 1, -1, -1, 1, 1, 1, -1, -1, -1, 1, -1, -1, -1, 1, -1, 1, 1, -1)
#X = rbind(r1, r2, r3, r4, r5, r6, r7, r8, r9, r10, r11, r12, r13, r14)

```
#E(s^2)-optimal, 26 factors, 12 runs
#r1 = c(-1, 1, 1, 1, -1, -1, -1, -1, -1, 1, 1, -1, -1, 1, 1, 1, -1, 1, -1, -1, 1, -1, 1, 1, -1, 1)
#r2 = c(-1, -1, -1, -1, 1, 1, 1, -1, -1, 1, 1, -1, -1, 1, 1, -1, 1, -1, -1, 1, -1, 1, 1, 1, 1, -1)
#r3 = c(1, 1, 1, 1, 1, -1, 1, -1, 1, -1, -1, 1, -1, 1, -1, 1, -1, -1, -1, -1, -1, 1, -1, 1, 1, -1)
#r4 = c(-1, 1, 1, -1, -1, 1, -1, 1, 1, -1, -1, -1, -1, -1, 1, 1, 1, -1, 1, -1, 1, 1, -1, -1, 1, 1)
#r5 = c(1, -1, -1, -1, 1, 1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1, -1, -1, 1, -1, -1, -1, -1, 1)
#r6 = c(-1, 1, 1, 1, -1, 1, 1, -1, -1, -1, -1, 1, 1, -1, -1, -1, 1, 1, 1, 1, -1, 1, 1, 1, -1, 1)
#r7 = c(-1, 1, -1, 1, 1, -1, -1, -1, 1, 1, 1, 1, 1, -1, 1, -1, 1, 1, 1, -1, -1, -1, -1, -1, 1, -1)
#r8 = c(-1, -1, -1, -1, -1, 1, -1, 1, 1, -1, 1, 1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, -1, -1, -1)
#r9 = c(1, -1, 1, -1, 1, -1, 1, 1, -1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, -1, 1, 1)
#r10 = c(1, -1, -1, 1, -1, -1, 1, 1, 1, 1, -1, -1, 1, 1, 1, -1, -1, -1, 1, -1, 1, 1, 1, -1, -1, -1)
#r11 = c(1, 1, -1, -1, 1, -1, -1, 1, 1, 1, -1, -1, 1, -1, -1, 1, 1, -1, -1, 1, -1, -1, 1, 1, -1, 1)
#r12 = c(1, -1, 1, 1, -1, 1, -1, 1, -1, 1, 1, 1, -1, -1, -1, -1, -1, -1, 1, 1, 1, -1, -1, 1, 1, -1)
#X = rbind(r1, r2, r3, r4, r5, r6, r7, r8, r9, r10, r11, r12)

n=nrow(X)
p=ncol(X)
nwant=1000
nrow=3*nwant
mynsis=n-1

BIC.SIS = matrix(rep(0), nrow, p)
BIC.ISIS = matrix(rep(0), nrow, p)
BIC.SISSCAD = matrix(rep(0), nrow, p)
BIC.ISISSCAD = matrix(rep(0), nrow, p)
BIC.SISSCAD.P = matrix(rep(0), nrow, p+1)
BIC.ISISSCAD.P = matrix(rep(0), nrow, p+1)

b = c(rep(0, p))
allcol = c(1:p)

#Model 1
#sigcol = sample(1:p,3,replace=F)
#coef = sample(c(-5,5),3,replace=T)

#Model 2(a)
sigcol = sample(1:p,4,replace=F)
coef = sample(c(-4,4),4,replace=T)

#Model 2(b)
#sigcol = sample(1:p,5,replace=F)
#coef = sample(c(-4,4),5,replace=T)
```

```
#Model 3
#sigcol = sample(1:p,6,replace=F)
#coef = sample(c(-3,3),6,replace=T)

#Model 4
#sigcol = sample(1:p,9,replace=F)
#coef = c(sample(c(-10,10),1), sample(c(-8,8),1), sample(c(-5,5),1), sample(c(-3,3),1), sample(c(-
2,2),5,replace=T))

numcol = length(sigcol)
nonsigcol = allcol[-sigcol]
b[sigcol[]] = coef[]

#set.seed(0)

#BIC Tuning Methods
for(i in 1:nrow)
{
BIC.SIS[i,]=0
BIC.ISIS[i,]=0
BIC.SISSCAD[i,]=0
BIC.ISISSCAD[i,]=0
BIC.SISSCAD.P[i,]=0
BIC.ISISSCAD.P[i,]=0
error = matrix(rnorm(n, mean=0, sd=1), n, 1)
y = X%*%b + error

BIC=try(SIS(data=list(X, y), model='glm', vartype=0, nsis=mynsis, family=gaussian(),
rank.method='coeff', eps0=1e-5, tune.method='BIC', post.tune.method='BIC', DOISIS=TRUE),
TRUE)

if (length(BIC)==1) {
BIC.SIS[i,]=NaN
BIC.ISIS[i,]=NaN
BIC.SISSCAD[i,]=NaN
BIC.ISISSCAD[i,]=NaN
BIC.SISSCAD.P[i,]=NaN
BIC.ISISSCAD.P[i,]=NaN }
```

```
else {
BIC.SIS[i,BIC$SISind]=1
BIC.ISIS[i,BIC$ISISind]=1
BIC.SISSCAD.P[i,] ¡- BIC$SIScoef
BIC.SISest = BIC$SIScoef[-1]
BIC.SISparam = abs(BIC.SISest) > 0
for(j in 1:p) {
if (BIC.SISparam[j]==TRUE) {
BIC.SISSCAD[i,j]=1
}
else {
BIC.SISSCAD[i,j]=0
}
}
BIC.ISISSCAD.P[i,] = BIC$ISIScoef
BIC.ISISest = BIC$ISIScoef[-1]
BIC.ISISparam = abs(BIC.ISISest) > 0
for(j in 1:p) {
if (BIC.ISISparam[j]==TRUE) {
BIC.ISISSCAD[i,j]=1
}
else {
BIC.ISISSCAD[i,j]=0
}
}
}
}

#One-step SIS important factors
BIC.SIS

#ISIS important factors
BIC.ISIS

#SIS-SCAD important factors
BIC.SISSCAD

#ISIS-SCAD important factors
BIC.ISISSCAD
```

```
#Getting rid of the NaN rows
BIC.SIS2 = BIC.SIS[BIC.SIS[,1]!="NaN",]
BIC.ISIS2 = BIC.ISIS[BIC.ISIS[,1]!="NaN",]
BIC.SISSCAD2 = BIC.SISSCAD[BIC.SISSCAD[,1]!="NaN",]
BIC.ISISSCAD2 = BIC.ISISSCAD[BIC.ISISSCAD[,1]!="NaN",]
BIC.SISSCAD.P2 = BIC.SISSCAD.P[BIC.SISSCAD.P[,1]!="NaN",]
BIC.ISISSCAD.P2 = BIC.ISISSCAD.P[BIC.ISISSCAD.P[,1]!="NaN",]

#Pulling the first "nwant" successful instances
BIC.SIS = BIC.SIS2[1:nwant,]
BIC.ISIS = BIC.ISIS2[1:nwant,]
BIC.SISSCAD = BIC.SISSCAD2[1:nwant,]
BIC.ISISSCAD = BIC.ISISSCAD2[1:nwant,]
BIC.SISSCAD.P = BIC.SISSCAD.P2[1:nwant,]
BIC.ISISSCAD.P = BIC.ISISSCAD.P2[1:nwant,]

#One-step SIS important factors
BIC.SIS

#ISIS important factors
BIC.ISIS

#SIS-SCAD important factors
BIC.SISSCAD

#ISIS-SCAD important factors
BIC.ISISSCAD

BIC.SIS.IR = matrix(rep(0), nwant, 4)
BIC.ISIS.IR = matrix(rep(0), nwant, 4)
BIC.SISSCAD.IR = matrix(rep(0), nwant, 4)
BIC.ISISSCAD.IR = matrix(rep(0), nwant, 4)
```

```
for (i in 1:nwant) {
if (sum(BIC.SIS[i,sigcol]) == numcol) {
BIC.SIS.IR[i,1]=1 }
else { BIC.SIS.IR[i,1]=0 }
if (sum(BIC.SIS[i,nonsigcol]) == 0) {
BIC.SIS.IR[i,2]=1 }
else { BIC.SIS.IR[i,2]=0 }
if (sum(BIC.SIS.IR[i,c(1:2)]) == 2) {
BIC.SIS.IR[i,3]=1 }
else { BIC.SIS.IR[i,3]=0 }
BIC.SIS.IR[i,4]=sum(BIC.SIS[i,])
}

for (i in 1:nwant) {
if (sum(BIC.ISIS[i,sigcol]) == numcol) {
BIC.ISIS.IR[i,1]=1 }
else { BIC.ISIS.IR[i,1]=0 }
if (sum(BIC.ISIS[i,nonsigcol]) == 0) {
BIC.ISIS.IR[i,2]=1 }
else { BIC.ISIS.IR[i,2]=0 }
if (sum(BIC.ISIS.IR[i,c(1:2)]) == 2) {
BIC.ISIS.IR[i,3]=1 }
else { BIC.ISIS.IR[i,3]=0 }
BIC.ISIS.IR[i,4]=sum(BIC.ISIS[i,])
}

for (i in 1:nwant) {
if (sum(BIC.SISSCAD[i,sigcol]) == numcol) {
BIC.SISSCAD.IR[i,1]=1 }
else { BIC.SISSCAD.IR[i,1]=0 }
if (sum(BIC.SISSCAD[i,nonsigcol]) == 0) {
BIC.SISSCAD.IR[i,2]=1 }
else { BIC.SISSCAD.IR[i,2]=0 }
if (sum(BIC.SISSCAD.IR[i,c(1:2)]) == 2) {
BIC.SISSCAD.IR[i,3]=1 }
else { BIC.SISSCAD.IR[i,3]=0 }
BIC.SISSCAD.IR[i,4]=sum(BIC.SISSCAD[i,])
}
```

```
for (i in 1:nwant) {
if (sum(BIC.ISISSCAD[i,sigcol]) == numcol) {
BIC.ISISSCAD.IR[i,1]=1 }
else { BIC.ISISSCAD.IR[i,1]=0 }
if (sum(BIC.ISISSCAD[i,nonsigcol]) == 0) {
BIC.ISISSCAD.IR[i,2]=1 }
else { BIC.ISISSCAD.IR[i,2]=0 }
if (sum(BIC.ISISSCAD.IR[i,c(1:2)]) == 2) {
BIC.ISISSCAD.IR[i,3]=1 }
else { BIC.ISISSCAD.IR[i,3]=0 }
BIC.ISISSCAD.IR[i,4]=sum(BIC.ISISSCAD[i,])
}

#IR = Identification Rate = % of simulations identifying only the active factors
(sum(BIC.SIS.IR[,3])/nwant)*100
(sum(BIC.ISIS.IR[,3])/nwant)*100
(sum(BIC.SISSCAD.IR[,3])/nwant)*100
(sum(BIC.ISISSCAD.IR[,3])/nwant)*100

#C = Coverage = % of simulations that identified all the active factors plus maybe others
(sum(BIC.SIS.IR[,1])/nwant)*100
(sum(BIC.ISIS.IR[,1])/nwant)*100
(sum(BIC.SISSCAD.IR[,1])/nwant)*100
(sum(BIC.ISISSCAD.IR[,1])/nwant)*100

#P = Power = Average % of active factors correctly declared as active
(sum(BIC.SIS[,sigcol])/(numcol*nwant))*100
(sum(BIC.ISIS[,sigcol])/(numcol*nwant))*100
(sum(BIC.SISSCAD[,sigcol])/(numcol*nwant))*100
(sum(BIC.ISISSCAD[,sigcol])/(numcol*nwant))*100

#T1 = Type 1 error rate = Average % of inactive factors incorrectly declared as active
(sum(BIC.SIS[,nonsigcol])/(length(nonsigcol)*nwant))*100
(sum(BIC.ISIS[,nonsigcol])/(length(nonsigcol)*nwant))*100
(sum(BIC.SISSCAD[,nonsigcol])/(length(nonsigcol)*nwant))*100
(sum(BIC.ISISSCAD[,nonsigcol])/(length(nonsigcol)*nwant))*100

#M = Mean # factors declared as active
sum(BIC.SIS.IR[,4])/nwant
sum(BIC.ISIS.IR[,4])/nwant
sum(BIC.SISSCAD.IR[,4])/nwant
sum(BIC.ISISSCAD.IR[,4])/nwant
```

```
#V = Variance of # factors declared as active
var(BIC.SIS.IR[,4])
var(BIC.ISIS.IR[,4])
var(BIC.SISSCAD.IR[,4])
var(BIC.ISISSCAD.IR[,4])

#Probability that a factor was identified as active
M = matrix(rep(0), p, 4)
for(i in 1:p) {
M[i,1] = sum(BIC.SIS[,i])/nwant*100
M[i,2] = sum(BIC.SISSCAD[,i])/nwant*100
M[i,3] = sum(BIC.ISIS[,i])/nwant*100
M[i,4] = sum(BIC.ISISSCAD[,i])/nwant*100 }

#Average parameter estimates
P = matrix(rep(0), p+1, 2)
for(i in 1:(p+1)) {
P[i,1] = mean(BIC.SISSCAD.P[,i])
P[i,2] = mean(BIC.ISISSCAD.P[,i]) }

M
P

#P(10,8) = Power = Model 4 Only, Dominant factors
#(sum(BIC.SIS[,c(4,15)])/(2*nwant))*100
#(sum(BIC.ISIS[,c(4,15)])/(2*nwant))*100
#(sum(BIC.SISSCAD[,c(4,15)])/(2*nwant))*100
#(sum(BIC.ISISSCAD[,c(4,15)])/(2*nwant))*100

#P(5,3) = Power = Model 4 Only, Moderate factors
#(sum(BIC.SIS[,c(2,13)])/(2*nwant))*100
#(sum(BIC.ISIS[,c(2,13)])/(2*nwant))*100
#(sum(BIC.SISSCAD[,c(2,13)])/(2*nwant))*100
#(sum(BIC.ISISSCAD[,c(2,13)])/(2*nwant))*100

#P(2) = Power = Model 4 Only, Small factors
#(sum(BIC.SIS[,c(1,8,11,17,26)])/(5*nwant))*100
#(sum(BIC.ISIS[,c(1,8,11,17,26)])/(5*nwant))*100
#(sum(BIC.SISSCAD[,c(1,8,11,17,26)])/(5*nwant))*100
#(sum(BIC.ISISSCAD[,c(1,8,11,17,26)])/(5*nwant))*100

b
```

Vita

Lindsey Beth Nicely was born on October 30, 1982 in Queens, New York. She moved to Virginia with her family when she was five years old and grew up in Midlothian, a suburb of Richmond. She graduated from the Chesterfield County Mathematics and Science High School in Midlothian, Virginia in 2000, and in 2004, she received her Bachelor of Science degree in Mathematics from Furman University in Greenville, South Carolina. She subsequently returned home to Richmond, Virginia and was hired by MeadWestvaco Corporation where she has worked for almost eight years in a variety of positions in Human Resources and Vendor Management. Lindsey is a candidate for a Master of Science degree in Mathematical Sciences with Statistics concentration at Virginia Commonwealth University in Richmond, Virginia and will graduate in May 2012. She lives in Glen Allen, Virginia with her husband, Ben, and their two papillons, Copper and Cammi.