The Pennsylvania State University

The Graduate School

NEW DEVELOPMENTS IN DESIGN OF EXPERIMENTS

A Dissertation in

Statistics

by

Jiayu Peng

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

December 2019

The dissertation of Jiayu Peng was reviewed and approved* by the following:

Dennis K.J. Lin
University Distinguished Professor of Supply Chain and Statistics
Dissertation Advisor, Chair of Committee

James L. Rosenberger
Emeritus Professor of Statistics, Director of SCC and Online Programs

Matthew Reimherr
Associate Professor of Statistics

Ethan X. Fang
Assistant Professor of Statistics

Russell R. Barton
Professor of Supply Chain and Information Systems and Industrial Engineering

Murali Haran
Department Head, Professor of Statistics

*Signatures are on file in the Graduate School.

Abstract

This dissertation consists of three projects on the design of experiments.

The first project is on order-of-addition experiments. In an order-of-addition problem, the output of a process depends on the order of adding different components into the system. The term "order-of-addition" originated from chemical experiments, in which the order of adding different reagents often affects the properties of final product. The importance of order-of-addition problem goes well beyond the area of chemistry though, for example, in the job scheduling field. An order-of-addition design is a set of pre-designed orders to test, so that the order effects can be most efficiently learned (in terms of some criteria), under certain experimental budget. In this project, we investigate the optimal design under different models of order effects, with the main results being: (1) a theory of D-optimal design under any order-of-addition linear model, and more specific theories under the pairwise-order model; (2) closed-form construction for a class of optimal and robust designs; (3) a fast and flexible algorithm for the construction of efficient designs with large number of components.

The second project is on the optimal design for estimating error variance. In screening experiments, the "saturated design with center points" is a conventional choice (in Response Surface Methodology). We propose an alternative design – the projection of a larger Hadamard matrix. In this project, we show that with certain assumption, the proposed design achieves the most efficient estimate of not only the factor effects but also the error variance, and is thus preferable over the conventional choice. Under various common error distributions, theories and simulation results are given to demonstrate that the proposed design achieves a more reliable estimate of error variance, as well as a reliable significance testing. For more practical use cases, the optimal follow-up design problem is studied. Relevant optimality theories are established and a convenient construction method for optimal follow-up design

is proposed.

The third project is about optimal run orders. In many industrial experiments, systematic run orders are desirable for reducing the costs from resetting factors, i.e., level changes. For special designs such as fractional factorials, minimum-level-change run orders are available in the literature. We propose an algorithm of solving optimal run orders for any arbitrary design and for flexible cost criteria. The algorithm applies modern Traveling Salesman Problem (TSP) devices. Specifically, the run order problem is converted into TSP and solved using a "Concorde" solver of TSP. Many new optimal run orders for a broad class of designs have been thus obtained.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First, I would like to express my deepest appreciation to my advisor, Dr. Dennis K.J. Lin. He contributed tremendous time, patience and effort in helping me going through the whole process of research. He taught me not only statistical knowledge but the philosophy of science. Under his guidance, I gradually grew from a math student to a real statistician and researcher. Even more importantly, he guided me through those difficult times in my PhD period, and drove me become mature. His whole-hearted effort for these 6 years has totally changed my life.

I would also like to sincerely thank Dr. Russel R. Barton, a great friend with whom I can discuss any wild idea on research. He guided me on the Swap-r algorithm research in Chapter 4 of this dissertation. In addition, he shared with me so many experiences and tips on career, academics, teaching, etc. Sincere thanks to Dr. James L. Rosenberger for his very detailed comments on my dissertation and oral defense. These comments substantially helped in improving the readability and strictness of my dissertation. Also, his kind support in my whole PhD period is appreciated. I would also like to thank Dr. Matthew Reimherr, and Dr. Ethan Xingyuan Fang for their insightful suggestions and kind support, as my teachers and committee members. The techniques I learned from them have been applied to my thesis research and also my current projects (in work). I would like to extend my sincere thanks to my collaborator, Dr. Mukerjee. I learned a great deal via his emails and detailed comments on our paper. His meticulous scholarship has influenced me a lot.

Of course, I want to thank the Department of Statistics at Penn State University for offering me the opportunity of graduate studies, and providing such a friendly, supportive and comfortable environment just like home. Last but not least, I would like to thank my parents: Mr. Derong Peng and Ms. Liping Yu for their unconditional love and support during the past twenty-nine years. I am so lucky to have grown up in this family. A special thanks to my girlfriend Xingyu Lu, who is always with me when I need her.

This work was supported National Science Foundation (NSF) via grant DMS 18102925. The content is solely the responsibility of the authors and does not necessarily represent the official views of NSF.

Chapter 1

Introduction

1.1. Overview of the three projects in this dissertation

Experiments are performed in all areas of science and industry. In the statistical sense, an experiment means a series of runs, where deliberate changes are made to the input variables, and the output variable is observed at each run. The design of experiments (DOE) studies how to control the levels of multiple input variables at each run. A good design could help experimenters collect more informative data, in the sense that the observations would result in valid and reliable conclusions at the later stage of analyzing the experiment. A good design can also substantially reduce the experimental cost and time. As DOE is more and more flexibly applied in different fields, a variety of new design problems have emerged. In this dissertation, we study three topics: (1) Optimal design of order-of-addition experiments, (2) Optimal design for error variance estimation and significance test, and (3) Construction of optimal run orders.

Optimal design of order-of-addition experiments

In an order-of-addition problem, the output of a process depends on the order of adding m different components into the system. The term "order-of-addition" originated from chemical experiments, in which the order of adding different reagents often affects the size, amount or purity of the final product. The importance of order-of-addition problem goes well beyond the area of chemistry though. For example, in the broad field of job scheduling, the manufacturing cost is often determined by the order of processing multiple jobs on the machine(s).

An order-of-addition problem aims to speculate optimal order(s) that maximizes or minimizes the response. For this purpose, an order-of-addition experiment compares the responses from multiple orders, to understand the dependence of response on order. As it is usually unaffordable to test all the $m!$ possible orders, the design problem arises to choose a subset of orders for comparison. This dissertation investigates the optimal design for estimation and prediction, under different models of order-of-addition effects. The main results are summarized as below.

1. For the general form of order-of-addition linear models, a theorem of D-optimal designs has been established via the approximate theory. This theorem paves the way for assessing the efficiency of any design under, to the best of our knowledge, all the order-of-addition models in the literature.

2. Under a specific class of models: pairwise order model with tapering, we present more theoretical results on design optimality and design construction. Explicit benchmarks for assessing design efficiency have been derived. Then, a class of optimal fractional designs has been obtained in closed forms, and these designs turn out to be robust to other models.

3. An algorithm has been developed to obtain efficient order-of-addition designs under any order-of-addition model, with (almost) any number of runs. The algorithm

exploits simulated annealing as well as a flexible choice of neighborhood size, to help escaping local optima. Update formulas are proposed to accelerate the computation; these formulas are new in the literature and extendable to other problems. Using the proposed algorithm, efficient and large-size (with m up to 50) designs have been obtained under the pairwise order model.

Optimal design for error variance estimation

With p effects to estimate, a saturated design means a design with the number of runs equaling the number of effects to estimate, i.e., model degrees of freedom. Saturated designs (such as Plackett-Burman designs) are economically favorable for screening experiments, but can be insufficient if the effect sparsity does not hold. As such, practitioners often add a few runs to a saturated design. Center points are the conventional choice of these additional runs.

In this project, we show that such convenient approach is not optimal, in terms of estimating the error variance. We propose alternative designs which achieves a minimum-variance and unbiased estimate of error variance. Both theoretical and experimental comparisons are shown to demonstrate the advantages of proposed designs.

Construction of optimal run orders

Many scientific and industrial experiments are carried out in a sequence. For an $n-$run design, there are $n!$ possible run orders to conduct the experiment. While the random run order is a default choice, some systematic run orders are actually desired and adopted in many experiments (Cox 1951, Draper and Stoneman 1968, Cheng and Steinberg 1991, etc.). Most often, systematic run orders are used either to minimize the cost of resetting factors, a.k.a. level changes, or to reduce the confoundedness between effect estimates and the time trend in the experimental process.

We are mainly interested in finding an optimal run order which minimizes the costs from level-changes. Cheng et al. (1998), Quinlan and Lin (2015) and other literature have developed the construction of minimum-level-change run orders for some popular designs such as factorial designs. However, these approaches are all limited to designs with specific structures. We propose an algorithm to construct optimal run orders for *any* design, with respect to flexible distance criterion. Our algorithm is based on the observation that the run order optimization is essentially a Travelling Salesman Problem (TSP). We illustrate how to convert the run order problem into a TSP with a modification, and how to solve the optimal run order using TSP solvers. A variety of new optimal run orders have been obtained in this way, and some of them are tabulated for practical use.

1.2. Organization

The remainder of my dissertation is organized as follows.

- Half of my dissertation (Chapters 2-4) is devoted to the first project, design of order-of-addition experiments.
  - Chapters 2 and 3 focus on the theory. As the order-of-addition design is a relatively new area in DOE, two fundamental questions remained unanswered before our work: how efficient can an order-of-addition design be? What kind of order-of-addition designs are optimal? Chapters 2 partially answered these two questions via the theory of D-optimal designs under a broad class of models. Chapter 3 gives more comprehensive answers to these questions under the pairwise order models, which are of particular interest in practice. The optimality theories in these two chapters depict

the upper bound in the efficiency of order-of-addition designs, and pave the way for future research in this area.

  - Chapter 4 focuses on the design construction. We propose a new algorithm for constructing order-of-addition designs with large number of components (m up to 50), while in the current literature, designs are available up to $m = 11$. Our new designs allow much wider application of order-of-addition designs, especially in computer experiments. This proposed algorithm has value in itself. It is a non-trivial extension of the classical exchange algorithm (e.g., Meyer and Nachtsheim 1995), and can be applied to other design problems as well.

- Chapter 5 is on the second project, optimal design for error variance estimation. This chapter proposes a class of optimal small screening designs as alternative to the conventional choices with "center points". Theoretical and empirical supports are given under different scenarios.

- Chapter 6 is on the third project, construction of optimal run orders. Given the choice of experimental runs, a generic algorithm is proposed to optimize the order of conducting these runs one by one. For practical use, a large number of explicit run orders are tabulated in this chapter. Our work here is also valuable in that it solves a classical design problem using modern devices in computer science and operations research.

- Chapter 7 highlights the main conclusions in my three projects, and describes some future work along the line of this dissertation.

This dissertation is a mixture of theoretically and practically oriented researches. In particular, Chapters 2, 3 and the second half of Chapter 5 are driven by some

theoretical questions that were never thoroughly answered in the literature; while the conclusions of these parts are important, the lengthy mathematics therein may not be interesting to all readers. One may skip these parts if s/he is mostly interested in explicit designs and relevant construction algorithms.

Chapter 2

Design of order-of-addition experiments: overview and general optimality theorem

2.1. Introduction

In an order-of-addition problem, the output of a process depends on the order of adding m different components into the system, and the interest is focused on understanding such dependence. Order-of-addition experiments have wide applications in, but not limited to, chemistry and related areas. For example, Song et al. (2014) conducted experiments under the 3! = 6 addition orders of the reagents CDs, Fe^{2+} and H_2O_2, and found that the [peak intensity of spectrum] for sequences Fe^{2+}→H_2O_2→CDs and H_2O_2→Fe^{2+}→CDs were obviously stronger than those for other sequences. More chemical order-of-addition experiments can be found in Fuleki and Francis (1968), Shinohara and Ogawa (1998), Ryberg (2008), and Ding et al. (2015), among others.

The importance of order-of-addition goes well beyond the area of chemistry. For example, optimizing the order is critical in the broad area of job scheduling, in which the cost of completing multiple jobs is often determined by the order of arranging these jobs. Job scheduling "plays an important role in most manufacturing and production systems as well as in most information processing environments" (Pinedo 2016).

With m components, it is often unaffordable to test all the $m!$ orders; for example, 10! is about 3.6 million. The design problem thus arises to choose a subset of orders for comparison. Primitive designs are often adopted in practice, as briefly surveyed below. Sometimes, a random design is adopted; for example, Stewart et al. (2001) randomly selected hundreds of orders to determine the optimal taxa order in a phylogenetic tree. In chemical experiments, the strategy of reversing pairs of reagents is often seen (Sugimoto et al. 1993 and Parikh et al. 2012), which is essentially a local search method. As for the aforementioned scheduling problem, except for some very simple scenarios, a large variety of scheduling problems do not permit a closed-form objective function. Therefore, one often has to apply heuristics to obtain optimal order(s); see, e.g., Pinedo (2016), Chapters 4-7 and 13-17. Most heuristics, such as simulated annealing and genetic algorithms, are essentially combining a random design on initial orders along with some local search methods.

Statistical design of experiments, which aims to improve the parameter estimation and prediction under a statistical model, has received successful application in many areas. Chapters 2-4 of this dissertation are devoted to the design of order-of-addition experiment along this line, motivated by the prevalence of order-of-addition experiment and the strong need to improve its design.

An early reference on the order-of-addition design is Van Nostrand (1995). He suggested considering a design to detect the pairwise order effects that are often

of much interest, with practitioners wishing to know, for example, if adding component 1 before 2 or vice versa has a significant influence on the response. The order-of-addition design starts receiving a great deal of attention from the statistical community, since Voelkel (2019). He studied design criteria and design construction under the pairwise order model of Van Nostrand (1995). A key feature of his work is the idea of extending orthogonal arrays to designs that are naturally restricted, of which the order-of-addition orthogonal array is an important example. He showed that such an array leads to the same value of D-criterion as the full design. Since Voelkel (2019), recent work including Zhao et al. (2019), Mee (2019) and Yang et al. (2019) further investigated the order-of-addition designs, with more models taken into consideration. While these papers obtained series of intuitively favorable designs, none of them, however, reported any optimality result.

This chapter establishes a general optimality result, which is anticipated but non-trivial to prove. We have shown that under any order-of-addition linear model with mild conditions, a design is D-optimal if and only if it has the same moment matrix as the full design. The result covers, to best of our knowledge, all the existing order-of-addition models. The optimality theory serves as a benchmark that paves the way for assessing the efficiency of any exact design under all different models. The remainder of this chapter is organized as below. Section 2.2 introduces the general form of order-of-addition linear model and three mild assumptions needed for the proposed theory. Section 2.3 introduces a number of specific models, under the umbrella of the general form in Section 2.2. Section 2.4 introduces preliminaries of optimal designs and approximate theory. The main result, Theorem 1, is then presented in Section 2.5. Its proof exploits Jensen's inequality as well as the fact that all the $m!$ orders form a group, i.e., the permutation group. Finally, Section 2.6 justifies that all these models satisfy the conditions in Section 2.2 and therefore are

covered by Theorem 1.

2.2. Order-of-addition linear models

In an order-of-addition experiment, suppose that there are m (≥ 3) components, labeled as $1, 2, \ldots, m$ respectively. An order of these components is a mapping

$$\begin{pmatrix} \text{Position} & 1 & 2 & \cdots & m \\ \text{Label} & a_1 & a_2 & \cdots & a_m \end{pmatrix},$$

where $a_1, \ldots, a_m$ form a permutation of $1, \ldots, m$. This order, denoted as $a = a_1 \cdots a_m$ for simplicity, means that the component labeled as a_j is at position j, i.e., the jth component to be added. Any such order is a treatment in the order-of-addition experiment, thus a total of $m!$ different treatments. Mathematically, these $m!$ treatments form a permutation group of order m, i.e., S_m. For any $a \in S_m$, write $\tau(a)$ as the mean of response arising from treatment a.

General Model Setup

We consider the mean regression problem which aims to describe and fit the mean function $\tau(\cdot)$. A realistic approach for approximating $\tau(\cdot)$ is to assume that an order determines the mean response through a number of *real-valued indicators*, that is, there exist functions $u : S_m \to \mathbb{R}^p$ and $g : \mathbb{R}^p \to \mathbb{R}$, p being any positive integer, such that $\tau(a) = g\big[u(a)\big]$ for any $a \in S_m$. As the function g can be approximated by its Taylor-series expansion, we further approximate $\tau(a)$ by a linear combination of multiple indicators of the order a. Explicitly, assume that for any $a \in S_m$,

$$\tau(a) = x'(a)\beta, \tag{2.1}$$

where $x : S_m \to \mathbb{R}^p$ is a function that can be flexibly chosen according to practical needs. Section 2.3 gives a number of models specified by different $x(a)$'s.

Model Assumptions

Throughout, we focus on the model satisfies the general form (2.1) and the assumptions A, B, C below.

Assumption A is the conventional Gauss-Markov assumption, i.e., the responses are uncorrelated and of equal variance. Admittedly, heteroskedasticity may exist in some order-of-addition problems. The optimal designs under the homoskedastic models, however, are useful for heteroskedastic models. For example, a D-optimal design under a heteroscedastic model can be constructed by modifying a D-optimal design under the corresponding homoskedastic model (Atkinson and Cook 1995, Sections 3.2 and 3.4).

Assumption B is that the model is estimable under the full design, that is, the $m! \times p$ model matrix $\left[x(a)\right]'_{a \in S_m}$ is of full rank.

Assumption C says that when the components are relabeled, the model can be reexpressed via a linear reparameterization. A relabeling is defined as a mapping

$$\begin{pmatrix} \text{Original label} & 1 & 2 & \cdots & m \\ \text{New label} & \pi_1 & \pi_2 & \cdots & \pi_m \end{pmatrix},$$

where $(\pi_1, \ldots, \pi_m)$ is a permutation of $(1, \ldots, m)$, i.e., $\pi = \pi_1 \cdots \pi_m \in S_m$. This mapping indicates that the component originally labeled as "j" is now labeled as "π_j" $(1 \le j \le m)$. Thus, after a relabeling $\pi = \pi_1 \cdots \pi_m$, an order $a = a_1 \cdots a_m$ is redenoted as $\pi_{a_1} \cdots \pi_{a_m}$ — this equals πa, i.e., the product of π and a in the permutation group.

Note that assumption C is naturally satisfied when an order-of-addition model is about all components rather than partial components in the sequence. As the

components, say, the chemical reagents are intrinsically not labeled, it should be allowable to freely relabel the components. Thus a model on treatment a should corresponds to a reparameterized model on πa. It is also natural to assume the linearity of reparameterization, which means that any addition of effects in the original model corresponds to an addition of effects in the reparametrized model.

Assumption C is mathematically described as below. For any $\pi \in S_m$, there exists a $p \times p$ matrix R_π, such that

$$x'(\pi a)\beta = x'(a)R'_\pi\beta \tag{2.2}$$

holds for any $a \in S_m$ and $\beta \in \mathbb{R}^p$. Both assumptions B and C are natural and mild restrictions on the form of $x(a)$. All the specific models in the next section satisfy the assumptions B and C.

All the assumptions here are mild: assumption A is commonly used in the literature, while B and C are natural restrictions of the form of $x(a)$ in (2.1). Existing order-of-addition models, as introduced in next section, all satisfy B and C.

2.3. Specific order-of-addition linear models

In this section, we specify a variety of order-of-addition models defined by giving explicit forms of $x(\cdot)$ in (2.1). Existing order-of-addition models, including the pairwise-order model of Van Nostrand (1995) and Voelkel (2019), the triplet-order model of Mee (2019) and the component-position model of Yang et al. (2019), are thus formulated. This section also incorporates some models proposed by ourselves, including the tapered models of Peng et al. (2019), and the models based on simple orders/positions as formulated in Lin and Peng (2019b).

2.3.1. Pairwise-order models with possible tapering

As in Van Nostrand (1995) and Voelkel (2019), the pairwise order model assumes that the variation in the response can be mostly explained by the precedence of each pair of components in the adding sequence. Here, we consider a more realistic, and broader class of pairwise order effect models with possible *tapering*, which allows the pairwise order to increase in the distance between the components in the pair. In these models, the impact of component i preceding j when they are at the two extremes of a sequence (e.g., i added first and j added last) can be less severe than that when they are next to each other — such tapering of effects is often seen in practice.

Explicitly, let S be the set of all pairs ij, $1 \le i < j \le m$. For $a = a_1 \cdots a_m \in S_m$ and $ij \in S$, write $h(ij, a)$ for the distance between i and j in a, that is, if $a_k = i$ and $a_l = j$, then $h(ij, a) = |k - l|$, so that $h(ij, a) \in \{1, \ldots, m-1\}$. The pairwise order model with tapering specifies that $p = m(m-1)/2 + 1$ and that

$$x(a) = \left[1, c_{h(ij,a)} \cdot z_{ij}(a)\right]'_{ij \in S}, \tag{2.3}$$

where for each $ij \in S$,

$$z_{ij}(a) = \begin{cases} 1 & \text{if } i \text{ precedes } j \text{ in } a \\ -1 & \text{if } j \text{ precedes } i \text{ in } a \end{cases} \tag{2.4}$$

In (2.3), the scalars c_h's with $0 \le c_{m-1} \le \cdots \le c_1 = 1$ are intended to capture the possible tapering as indicated above. For example, one can take $c_h = 1/h$ or $c_h = c^{h-1}$, with known c, $0 < c < 1$. On the other hand, if no such tapering is anticipated then one can set $c_h = 1$ for all h, in which case (2.3) and (2.4) get reduced to the usual

pairwise order model of Van Nostrand (1995) and Voelkel (2019).

Example 1. As an illustration, when $m = 4$, a pairwise order model with tapering c^h assumes that the expectation of response from order a equals

$$\tau(a) = \beta_{12} \cdot z_{12}(a) + \beta_{13} \cdot c \cdot z_{13}(a) + \beta_{14} \cdot c^2 \cdot z_{14}(a) + \beta_{23} \cdot \cdot z_{23}(a) + \\ \beta_{24} \cdot c \cdot z_{24}(a) + \beta_{24} \cdot z_{34}(a),$$

where $z_{ij}(a)$'s are the pairwise order indicators in (2.4), β_{ij}'s are the parameters to estimate, and $c \in (0, 1)$ is a pre-specified constant or a tuning parameter (determined by cross validation).

2.3.2. Triplet-order model

Beyond the pairwise order model, it is sometimes desirable to further capture the impact of the precedence pattern among triplets, rather than pairs of components.

The conventional wisdom of constructing a higher-order model from a lower-order model is to add interactions. Intuitively, the ordering pattern of a triplet i, j, k $(1 \le i < j < k \le m)$ can be represented by the pairwise order indicators $z_{ij}(a)$, $z_{ik}(a)$, $z_{jk}(a)$ as well as their two-way interactions, i.e., $z_{ij}(a)z_{ik}(a)$, $z_{ij}(a)z_{jk}(a)$ and $z_{ik}(a)z_{jk}(a)$. Mee (2019) noted that a model including all these terms along with the intercept, however, leads to a singular model matrix even under the full design, due to the fact that $z_{ij}(a)z_{ik}(a) - z_{ij}(a)z_{jk}(a) + z_{ik}(a)z_{jk}(a) = 1$ for any $a \in S_m$. An estimable model can be defined by removing one interaction term for each triplet. For instance, by removing $z_{ik}(a)z_{jk}(a)$, Mee (2019) defines a triplet-order model with

$$x(a) = \left[1, z'(a), t_1'(a), t_2'(a)\right]', \tag{2.5}$$

where the vectors

$$z(a) = \left[z_{ij}(a)\right], t_1(a) = \left[z_{ij}(a)z_{ik}(a)\right]_{1\le i<j<k\le m} \text{ and } t_2(a) = \left[z_{ij}(a)z_{jk}(a)\right]_{1\le i<j<k\le m}.$$

Example 2. When $m = 4$, Mee's triplet-order model assumes that the expected response is

$$\begin{aligned}\tau(a) = {} & \beta_{12}z_{12}(a) + \beta_{13}z_{13}(a) + \beta_{14}z_{14}(a) + \beta_{23}z_{23}(a) + \beta_{24}z_{24}(a) + \beta_{24}\cdot z_{34}(a) + \\ & \beta^{(1)}_{123}z_{12}(a)z_{13}(a) + \beta^{(2)}_{123}z_{12}(a)z_{23}(a) + \beta^{(1)}_{124}z_{12}(a)z_{14}(a) + \beta^{(2)}_{124}z_{12}(a)z_{24}(a) + \\ & \beta^{(1)}_{134}z_{13}(a)z_{14}(a) + \beta^{(2)}_{134}z_{13}(a)z_{34}(a) + \beta^{(1)}_{234}z_{23}(a)z_{24}(a) + \beta^{(2)}_{234}z_{23}(a)z_{34}(a),\end{aligned}$$

where $\beta_{ij}, \beta^{(1)}_{ijk}, \beta^{(2)}_{ijk}$'s are parameters to estimate.

Note that the triplet order model can be also extended to a tapered version: just multiply each z_{ij} by c_h in (2.3).

2.3.3. Component-position model

Yang et al. (2019) proposed a model to determine "which component should be added in each specific position". Under the general setup (2.1), their model specifies $x(a) = \left[1, u_{ij}(a)\right]'_{2\le i,j\le m}$, where

$$u_{ij}(a) = \begin{cases} 1 & \text{if } a_i = j \\ 0 & \text{if } a_i \neq j \end{cases} \tag{2.6}$$

for any $1 \le i, j \le m$. That is, $u_{ij}(a)$ indicates whether component j is at position i in treatment a. In view of the restrictions $\sum_{i=1}^{m} u_{ij}(a) = 1$ $(1 \le j \le m)$ and $\sum_{j=1}^{m} u_{ij}(a) = 1$ $(1 \le i \le m)$, $x(a)$ does not includes any $u_{1j}(a)$'s nor $u_{i1}(a)$'s in order that the model

is estimable. See Yang et al. (2019) for details. As they mentioned, this model gives rise to a one-way ANOVA to test if placing different components at a specific position significantly affects the response.

2.3.4. Model based on simple orders or ranks

If we simply view $a = a_1 \cdots a_m \in S_m$ as a vector $a = [a_1, \cdots, a_m]'$ in $\mathbb{R}^m$, it is plausible to approximate the mean function $\tau(a)$ by polynomials in $a_1, \ldots a_m$, for instance, the Taylor expansions of $\tau(a)$. This idea has been implicitly suggested in Anderson-Cook and Lu (2019). The label a_j is called a "simple order". A first-order model based on simple orders can be defined by specifying

$$x(a) = (1, a_1, \ldots, a_{m-1})'. \tag{2.7}$$

Note that a_m is not included here, due to the constraint $\sum_{j=1}^m a_j = \sum_{j=1}^m j = m(m+1)/2$. A second model includes linear, quadratic, as well as two-way interactions of a_j's. Thus in view of (2.7), a second-order model based on simple orders is intuitively defined by

$$x(a) = \left(1, a_j, a_j^2, a_j a_k\right)'_{1 \le j < k \le m-1}.$$

However, the above equation turns out to result in an unidentifiable model. It actually contains 1 extra degree of freedom, due to an non-obvious constraint

$$\sum_{1 \le i < j \le m-1} a_i a_j - C_1 \sum_{1 \le j \le m-1} a_j = C_2/2, \tag{2.8}$$

where the constants $C_1 = \sum_{j=1}^{m} j = m(m+1)/2$ and $C_2 = \sum_{j=1}^{m} j^2 = m(m+1)(2m+1)/6$. By removing, say, the last interaction, the resulting model with

$$x(a) = \left(1, a_j, a_j^2, a_j a_k\right)'_{1 \le j < k \le m-1, j \ne m-2}. \tag{2.9}$$

becomes estimable.

For the purpose of interpretation, it is sometimes favorable to model the reliance of the response on the rank (i.e., position) of each component. Let r_j be the position of component j, i.e., $r_j = k$ if and only if $a_k = j$. Analogic to the above equations (2.7) and (2.9), a first-order model on ranks can be defined by

$$x(a) = (1, r_1, \ldots, r_{m-1})', \tag{2.10}$$

whereas a second-order model on ranks can be defined by

$$x(a) = \left(1, r_j, r_j^2, r_j r_k\right)'_{1 \le j < k \le m-1, j \ne m-2}. \tag{2.11}$$

As mentioned in the last section, all the models in these section satisfies assumptions B and C. To check assumption B for any particular model, it suffices to find a set of $a^{(1)}, \ldots, a^{(p)} \in S_m$ such that the $p \times p$ matrix $\left[x(a^{(1)}), \ldots, x(a^{(p)})\right]'$ is non-singular. Under any model herein, such $a^{(1)}, \ldots, a^{(p)}$ can be easily found by random search, for a specific m. It will be shown in Section 2.8, under a general framework, why these models satisfy assumption C.

2.4. Preliminary: optimal designs

Consider any n-run order-of-addition experiment. In practice, n is often specified according to the experimental budget. Experimenters will choose n treatments, i.e., orders $a^{(1)}, \ldots, a^{(n)}$ to test, and observe a response y_i, respectively, from each treatment $a^{(i)}$. Under model (2.1), the least squares estimator of β is

$$\hat{\beta} = (X'X)^{-1}X'y,$$

where $X = \left[x(a^{(1)}), \ldots, x(a^{(n)})\right]'$ is and $y = (y_1, \ldots, y_n)'$. Recall that the $x(\cdot)$ is any function from S_m to $\mathbb{R}^p$ with any $p \geq 1$. Under the Gaussian-Markov assumption that the responses are uncorrelated and have equal variance σ^2 for different a's, $\hat{\beta}$ is unbiased and has a variance-covariance matrix of

$$\mathrm{Var}(\hat{\beta}) = \sigma^2(X'X)^{-1}. \tag{2.12}$$

The estimated value for the mean response from any order a is $\hat{\tau}(a) = x'(a)\hat{\beta}$, which is unbiased with a variance of

$$\mathrm{Var}\left[\hat{\tau}(a)\right] = \sigma^2 x'(a)(X'X)^{-1}x(a). \tag{2.13}$$

The efficiency of $\hat{\beta}$ depends on the properties of the model matrix X, which are determined by the choice $a_1, \ldots, a_n$. The optimal design problem consists of choosing $a_1, \ldots, a_n \in S_m$ under some sensible optimality criteria. Equations (2.12) and (2.13) motivate the definitions of various design criteria as introduced below.

2.4.1. Design criteria

One of the first to state such a criterion was Smith (1918). She proposed to minimizing the maximum variance of any predicted value, namely,

$$\underset{a_1,\ldots,a_n \in S_m}{\arg\min} \max_{a \in S_m} \mathrm{Var}\big[\hat{\tau}(a)\big],$$

which was later called the global, or G-optimality in Kiefer and Wolfowitz (1959). Being a criterion regarding the prediction errors, the G-optimality is worth consideration for the purpose of finding (nearly)-optimal orders.

Most design criteria proposed later on put emphasis on the precision of estimating β. These criteria minimize $\mathrm{Var}(\hat{\beta})$ in some senses. As $\mathrm{Var}(\hat{\beta})$ of (2.12) is inversely proportional to $M = X'X/n$, these criteria maximize $\phi(M)$ for certain functions ϕ's. Popular choices for ϕ include:

- D-criterion (Wald 1943): $\phi(M) = \log\big(|M|\big)$, where $|\cdot|$ denotes the determinant.
- A-criterion (Chernoff 1953): $\phi(M) = -\operatorname{tr}(M^{-1})$, where $\operatorname{tr}(\cdot)$ denotes the trace.
- E-criterion (Ehrenfeld 1955): $\phi(M)$ takes the minimum eigenvalue of M.
- M.S.-criterion (Eccleston and Hedayat 1974): $\phi(M) = -\operatorname{tr}(M^2)$.

The D-optimality appears to be the most frequently used in the literature. One of its unique advantages is that it is invariant under reparameterization (Pukelsheim 2006, page 136). It is also well-interpreted: a D-optimal design minimizes the volume of the confidence ellipsoid of $\hat{\beta}$ at any confidence level (Kiefer and Wolfowitz 1959). Another important property of D-optimality, established in Kiefer and Wolfowitz (1960), is its equivalence to the G-optimality in the approximation theory.

We close this section by the following remark. In design of experiments, one usually considers the per-observation moment matrix $X'X/n$ instead of the usual moment matrix $X'X$. This is because using $X'X/n$ allows us to assess the efficiencies of designs with different n's, relative to a single benchmark. Throughout, we call $M = X'X/n$ as a moment matrix for simplicity, following classical textbooks such as Pukelsheim (2006).

2.4.2. Approximate design theory

Here I clarify the difference between an exact design and a design measure. Any exact design is a collection of treatments. Explicitly, in an N-run exact design d, any treatment $a \in S_m$ is replicated $r(a)$ times, such that the integers $r(a)$ (≥ 0) sum to N. By (2.1), d has per run moment matrix

$$M(w) = \sum_{a \in S_m} w(a)x(a)x(a)', \tag{2.14}$$

where $w(a) = r(a)/N$. In approximate theory, the requirement on the design weights $w(a)$ to be integral multiples of $1/N$ is relaxed, and the $w(a)$ are allowed to be any nonnegative quantities subject to $\sum_{a \in S_m} w(a) = 1$. Then $w = \{w(a) : a \in S_m\}$ is called a design measure, having moment matrix $M(w)$ in (2.14).

By definition, any exact design well corresponds to a design measure. In particular, the full design d_0 corresponds to the uniform design measure w_0 over S_m, which has every $w(a) = 1/m!$ and therefore a moment matrix

$$M_0 = M(w_0) = (1/m!) \sum_{a \in S_m} x(a)x(a)'. \tag{2.15}$$

An arbitrary design measure, however, does not necessarily correspond to an exact

design.

The use of design measure will facilitate our optimality theory in the next section. Note that under any optimality measure, if any exact design corresponds to an optimal design measure, then the exact design itself must be optimal among all the exact designs of the same number of runs.

2.5. Main result

Let d_0 be the full order-of-addition design which replicates each order once. Theorem 1 in this Section establishes the D-optimality of d_0 for inference on β under a broad range of criteria, among all designs with the same number, $m!$, of runs. This result is rather strong because it holds irrespective of the form of $x(a)$ in (2.1).

One may anticipate the optimality of d_0 because, with run size $m!$, a design that replicates some treatments more than once while omitting some others altogether is intuitively unappealing. But, unlike in similar situations like traditional full factorials, the moment matrix of d_0 is rather involved. For instance, it needs not to be completely symmetric (Kiefer 1975) in the sense of having all diagonal elements equal as well as all off-diagonal elements equal. Different forms of $x(\cdot)$ further complicate matters. See Section 2.6 for examples of such complicated moment matrices, under different order-of-addition models. As a result, a direct combinatorial proof of the optimality of d_0 is quite challenging. An approach, based on the approximate theory, is found to yield a subtle noncomputational proof that does not even require explicit evaluation of the moment matrix of d_0.

Theorem 1. *Given assumptions A, B and C given in Section 2.2, a design measure w is D-optimal under model (2.1) if and only if $M(w) = M_0$.*

The proof of Theorem 1 exploits the strict concavity of $\phi(\cdot)$, the interchangeability

between determinant and matrix multiplication, as well as the following basic result in the group theory.

Lemma 1. *For any $\pi \in S_m$ and any $n \geq 1$, let π^n be the product of n π's under the group multiplication rule. Then $\pi^{m!}$ equals the unity in S_m, i.e., $a\pi^{m!} = \pi^{m!}a = a$ for any $a \in S_m$.*

Lemma 1 holds because any element in a group generates a cyclic subgroup, of which the order divides the order of the whole group; its explicit proof is skipped for simplicity. Lemma 1 leads to the following observation which plays a key role in the proof of Theorem 1.

Lemma 2. *Consider any model in the form equation (2.1) that satisfies assumptions B and C. Then for any $\pi \in S_m$, the matrix R_π in assumption C must have a determinant of either 1 or -1.*

Proof. By assumption B, there exist $a_1, \ldots, a_p \in S_m$ such that $X = \left[x'(a_1), \ldots, x'(a_p)\right]'$ is a non-singular squared matrix. By assumption C and mathematical induction, $x'(\pi^{m!}a_i)R_\pi^{m!} = x'(a_i)$ for any $1 \leq i \leq p$. By Lemma 1, $a_i\pi^{m!} = a_i$, and thus $x'(a_i)A_\pi^{m!} = x'(a_i)$ for any $1 \leq i \leq p$, which implies that $XR_\pi^{m!} = X$. Since X is non-singular, $R_\pi^{m!}$ equals the identity matrix. Hence $|R_\pi| = 1$ or -1. □

We now complete the proof of Theorem 1. By assumption B, M_0 is positive-definite. We can suppose that $M(w)$ is also positive-definite, otherwise $\left|M(w)\right| < \left|M_0\right|$. Consider any $\pi \in S_m$: by assumption C, $x'(\pi a)\beta = x'(a)R_\pi'\beta$ for any $a \in S_m$ and $\beta \in \mathbb{R}^p$, and therefore $x'(\pi a) = x'(a)R_\pi'$ for any $a \in S_m$. Thus

$$M(\pi w) = \sum_{a \in S_m} w(a)R_\pi x(a)x'(a)R_\pi' = R_\pi' M(w)R_\pi. \tag{2.16}$$

Recall that the D-criterion corresponds to $\phi(M) = \log\big(|M|\big)$. Hence by (2.16) and Lemma 2,

$$\phi\big[M(\pi w)\big] = \phi\big[M(w)\big] \tag{2.17}$$

for any $\pi \in S_m$ and design measure w.

Since $\phi(\cdot)$ is strictly concave on the manifold of all the $p \times p$ positive definite matrices, by Jensen's inequality,

$$(1/m!) \sum_{\pi \in S_m} \phi\big[M(\pi w)\big] \leq \phi\Big[(1/m!) \sum_{\pi \in S_m} M(\pi w)\Big], \tag{2.18}$$

with the equality holding if and only for any $\pi \in S_m$,

$$M(\pi w) = (1/m!) \sum_{\pi \in S_m} M(\pi w). \tag{2.19}$$

Note that $(1/m!) \sum_{\pi \in S_m} M(\pi w) = (1/m!) \sum_{\pi \in S_m} \sum_{a \in S_m} w(a) x(\pi a) x'(\pi a) = (1/m!)$ $\sum_{a \in S_m} w(a) \cdot \sum_{\pi \in S_m} x(\pi a) x'(\pi a)$. Since $\{\pi a : \pi \in S_m\} = S_m$ and that $\sum_{a \in S_m} w(a) = 1$, we have

$$(1/m!) \sum_{\pi \in S_m} M(\pi w) = (1/m!) \sum_{a \in S_m} x(a) x'(a) = M_0.$$

On the other hand, by (2.17), $\phi\big[M(\pi w)\big] = \phi\big[M(w)\big]$ for any $\pi \in S_m$. Thus in view of equations (2.18) and (2.19), $\phi\big[M(w)\big] \leq \phi(M_0)$, with the equality holding if and only if $M(\pi w) = M_0$ for any $\pi \in S_m$, and the result follows.

The knowledge of optimal designs is useful in practice. For example, suppose one has already obtained a design with D-criterion $|M|^{1/p} = 0.56$. If we know theoretically that a D-optimal design has $|M|^{1/p} = 0.6$, then the obtained design is quite favorable. If, on the other hand, a D-optimal design has $|M|^{1/p} = 0.8$, then there will be a need to further improve the obtained design. In view of this, the optimality Theorem 1 paves

the way for assessing the D-efficiency of any exact design, under any order-of-addition linear model that satisfies the mild assumptions A, B and C.

Our ultimate goal, as described in Section 2.1, is to obtain robust designs that are efficient under a variety of order-of-addition models. Theorem 1 is an important intermediate step for exploring the design robustness, since it is applicable to various models, covering all those in Section 2.3.

Remark 1 (Optimality of full design does not hold under other criteria)**.** We make a note that the optimality Theorem 1 only holds for the D-criterion. Under a general order-of-addition linear model, the full design need not be optimal under the A-, E-, or MS-criterion. As an example, consider the triplet-order model (2.5) with $m = 6$. Let $\tilde{d}$ be the design obtained by removing the order 153264, while replicating the order 123456 once, from the full design d_0. It turns out that $\tilde{d}$ has $-\operatorname{tr}(M^{-1}) = -132.96 > \operatorname{tr}(M_0^{-1}) = -133.08$, $\lambda_{\min}(M) = 0.108 > \lambda_{\min}(M_0) = 0.105$ and $-\operatorname{tr}(M^2) = -128.0804 > -\operatorname{tr}(M_0^2) = -128.0810$. Thus literally, $\tilde{d}$ is more favorable than d_0 under any of the A-, E- and MS-criteria. It is a non-trivial and perhaps surprising result that the full design is always optimal under the D-criterion.

2.6. More theoretical supports

As aforementioned, all the models in Section 2.3 satisfy the assumptions B and C (see Section 2.2), and thus the D-optimality theorem can be applied. This statement should be verified. Here we illustrate why the assumptions are satisfied in these models.

Assumption B, "non-singularity of the full model matrix" — this can be checked numerically by evaluating the determinant of full model matrix, given any specific m. We've numerically verified this assumption for all the models in Section 2.3, with m

from 4 up 30. Details are omitted here, since this validation can be easily repeated.

Assumption C – "relabeling invariance" – shall be theoretically verified. Here, we introduce a general theory to verify assumption C for all different models, instead of justifying it from model to model.

As can be seen from Section 2.3, the establishment of an order-of-addition linear model often starts from considering a class of candidate factors. For example, when Mee (2019) studies the definition of a triplet-order model, the candidate factors he considered include all the pairwise-order indicators as well as their two-way interactions. A model including *all* the candidate factors, however, is likely to be unidentifiable. In the example of triplet-order model, the interactions of pairwise-order indicators are confounded with each other (and the intercept). Thus, a natural strategy of defining an identifiable model is to incorporate only a *subset* of candidate factors into the model. This strategy was adopted in Mee (2019), Yang et al. (2019) and Lin and Peng (2019b),

The concept of candidate factors facilitates the verification of assumption C for the models in Section 2.3. Call the model incorporating all the candidate factors as a candidate model. We found that it is straightforward to verify assumption C for candidate models; moreover, if the candidate model satisfies assumption C, then any model incorporating a full-rank subset of candidate factors satisfies assumption C as well. Mathematical descriptions are given below.

Let $\tilde{\mathcal{V}} = \{x_j(\cdot)\}_{1\le j\le \tilde{p}}$ be a class of candidate factors. As mentioned, the $m! \times \tilde{p}$ matrix $\left[x_j(a)\right]_{a\in S_m, 1\le j\le \tilde{p}}$ is often singular. In this case, one would choose a subset $S \subset \{1,\ldots,\tilde{p}\}$ with $|S| = p$, such that the $m! \times p$ matrix $\left[x_j(a)\right]_{a\in S_m, j\in S}$ is of full rank, whereas for any $j^* \in \{1,\ldots,\tilde{p}\}\backslash S$, the $m!\times(p+1)$ matrix $\left[x_j(a)\right]_{a\in S_m, j\in S\cup\{j^*\}}$ becomes singular. We call $\mathcal{V} = \{x_j(\cdot)\}_{j\in S}$ as a full-rank subset of the candidate set $\tilde{\mathcal{V}}$, and p as the degrees of freedom of the candidate set. In other words, p is the rank of the

matrix $\left[x_j(a)\right]_{a\in S_m, 1\le j\le \tilde{p}}$.

Theorem 2. *Consider an order-of-addition model in the form of (2.1) that incorporates any candidate set $\tilde{\mathcal{V}}$. Suppose that this model satisfies assumption C. Then any model incorporating a full-rank subset of $\tilde{\mathcal{V}}$ satisfies both assumptions B and C.*

Proof. Recall that assumption C holds if and only if for any $\pi \in S_m$, there exists a $p \times p$ matrix R_π such that (2.2) holds for any $a \in S_m$ and $\beta \in \mathbb{R}^p$, or equivalently, $x(\pi a) = R_\pi x(a)$ holds for any $a \in S_m$.

Let the candidate set be $\tilde{\mathcal{V}}\{x_j(\cdot)\}_{1\le j\le \tilde{p}}$, and $\tilde{x}(a) = [x_j(\cdot)(a)]_{1\le j\le \tilde{p}}$. Since the model incorporating the whole candidate set satisfies assumption C, for any $\pi \in S_m$, there exists a $\tilde{p} \times \tilde{p}$ matrix $\tilde{R}_\pi$ such that

$$\tilde{x}(\pi a) = \tilde{R}_\pi \tilde{x}(a) \tag{2.20}$$

holds for any $a \in S_m$. Let $\mathcal{V} = \{x_j(\cdot)\}_{j\in S}$ be any full-rank subset of $\tilde{\mathcal{V}}$. Let $x(a) = [x_j(a)]_{j\in S}$, then obviously, there exists a matrix P such that $x(a) = P\tilde{x}(a)$ for any $a \in S_m$. On the other hand, since $\mathcal{V}$ is a full-rank subset, every column of $[x_j(a)]_{a\in S_m, 1\le j\le \tilde{p}}$ can be expanded by a linear combination of the columns of $[x_j(a)]_{a\in S_m, j\in S}$. Consequently, there exists a $\tilde{p} \times p$ matrix Q, such that $\tilde{x}(a) = Qx(a)$ for any $a \in S_m$. Thus by (2.20), for any $\pi \in S_m$, we have

$$x(a) = P\tilde{R}_\pi Q x(a)$$

for any $a \in S_m$. Hence the model with model point $x(\cdot)$ satisfies assumption C as well. Meanwhile, it apparently satisfies assumption B. □

All the models in Section 2.3 are obtained via selecting a full-rank subset from a candidate set. (For the pairwise-order model in Section 2.3.1, the candidate model

itself is of full rank.) To show that all these models satisify assumptions B and C, it suffices to show that the candidate model of these models satisfy assumption C. The candidate set is

$$\tilde{\mathcal{V}}_0 = \{z_{ij}(\cdot)\}_{1\le i<j\le m}$$

for the pairwise-order modelwith the $z_{ij}(\cdot)$ defined by equation (2.3); is

$$\tilde{\mathcal{V}}_1 = \{z_{ij}(\cdot)\}_{1\le i<j\le m} \cup \{z_{ij}(\cdot)z_{ik}(\cdot), z_{ij}(\cdot)z_{jk}(\cdot), z_{ik}(\cdot)z_{jk}(\cdot)\}_{1\le i<j<k\le m}$$

for the triplet-order model (2.5); is

$$\tilde{\mathcal{V}}_2 = \{u_{ij}(\cdot)\}_{1\le i,j\le m}$$

for the component-position model, with the $u_{ij}(\cdot)$ defined by equation (2.6); is

$$\tilde{\mathcal{V}}_3 = \{a_i\}_{1\le i\le m} \text{ and } \tilde{\mathcal{V}}_4 = \{a_i a_i^2\}_{1\le i\le m}$$

for the simple-order-based models defined by (2.7) and (2.9) respectively; and is

$$\tilde{\mathcal{V}}_5 = \{r_i(a)\}_{1\le i\le m} \text{ and } \tilde{\mathcal{V}}_6 = \{r_i(a)r_i^2(a)\}_{1\le i\le m},$$

for the rank-based models defined by (2.10) and (2.11) respectively. We have:

Lemma 3. *Let $\tilde{\mathcal{V}}$ be any of the above candidate sets, i.e., $\tilde{\mathcal{V}}$ $(0 \le j \le 5)$, and π be any element in S_m. Then there exists a mapping $\pi^* : \tilde{\mathcal{V}} \to \tilde{\mathcal{V}}$, such that for any $v(\cdot) \in \tilde{\mathcal{V}}$, either $v(\pi a) = \pi^* v(a)$ for any $a \in S_m$, or $v(\pi a) = -\pi^* v(a)$ for any $a \in S_m$.*

Proof. Consider $\tilde{\mathcal{V}}_0 = \{z_{ij}(\cdot)\}_{1\le i<j\le m}$. Given any $\pi = \pi_1 \cdots \pi_m$, for $1 \le i < j \le m$, let $\bar{\pi}_i\bar{\pi}_j$ equal $\pi_i\pi_j$ if $\pi_i < \pi_j$, and $\pi_j\pi_i$ if $\pi_i > \pi_j$. Then, for every $1 \le i < j \le m$ and

every $a \in S_m$, (i) i precedes j in a if and only if π_i precedes π_j in πa, and (ii) the distance between i and j in a equals that between π_i and π_j in πa. Recalling (2.4), therefore, by (i), $z_{\bar{\pi}_i\bar{\pi}_j}(\pi a)$ and $z_{ij}(a)$ have the same sign if and only if $\pi_i < \pi_j$, while by (ii), they have the same absolute value. In other words, $z_{\bar{\pi}_i\bar{\pi}_j}(\pi a)$ equals $z_{ij}(a)$ if $\pi_i < \pi_j$, and $-z_{ij}(a)$ if $\pi_i > \pi_j$, and this completes the proof for $\tilde{\mathcal{V}}_0$. For other $\tilde{\mathcal{V}}_j$'s, the conclusion can be proved likewise. □

Lemma 3 implies that all the candidate models considered here satisfy assumption C, or explicitly:

Corollary 1. *Let $\tilde{\mathcal{V}}$ be any of the $\tilde{\mathcal{V}}_j$ ($0 \le j \le 5$). Let $y(a) = x'(a)\beta$ be the candidate model given by the candidate set $\tilde{\mathcal{V}}$. Then given any $\pi \in S_m$, there exists a signed permutation matrix $R(\pi)$, such that $x'(\pi a) = x'(a)R(\pi)$, for every $a \in S_m$.*

Recall that a signed permutation matrix is a square matrix having exactly one nonzero entry in each row and column, where any nonzero entry is 1 or −1. In view of 1, the models given in Section 2.3 all satisfy assumptions B and C. Up to here, we have completely justified the main take-way of this chapter: under the classical Gauss-Markov assumption, the full design is D-optimal for the five types of order-of-addition models in Section 2.3: pairwise-order, triplet-order, component-position, order-based, and rank-based models.

Chapter 3

Design of order-of-addition experiments under the pairwise-order model

Theorem 1 established in the last chapter is applicable to various order-of-addition models, including all the models in Section 2.3. While the design problem under all these models deserve more investigation, this chapter focuses on the designs under the pairwise order model (defined in Section 2.3.1). Our ultimate goal is to obtain designs that are efficient under multiple models. To achieve this, a convenient approach is to first find a class of optimal designs under one model, and then select, from this class, designs that are robust to other models. Following this strategy, we first obtained a broad class of optimal fractional designs under the pairwise order model, and among them, obtained robust ones under other models. These are the main results of this chapter, presented in Section 3.2.

Other sections of this chapter are theoretical supports of the main results above. Section 3.1 establish some results that further facilitates assessing the efficiency of any

pairwise order design. Section 3.3 includes the proofs for all the results in Sections 3.1 and 3.2.

3.1. Theory of optimal pairwise-order designs

Under a general linear model (2.1) with mild assumptions, the full order-of-addition design attains the D-optimality. It need not be optimal under other, say, the A-, E- and MS-criteria, as shown by the counter-examples in Section 2.6. Under the pairwise-order of model, however, the optimality of full design holds for a very broad range of criteria, as indicated by the result below.

Theorem 3. *(a) Let $\phi(M)$ be any concave design criterion that can be fully determined by the eigenvalues of M. Consider any model in form (2.1) that satisfies the assumptions A, B, and C given in Section 2.2. Suppose that for this model, the matrix R_π in assumption C is an orthogonal matrix. Then under this model, the uniform design measure w_0 is ϕ-optimal. If ϕ is strictly concave, then any design measure w is ϕ-optimal if and only if $M(w) = M_0$.*

(b) For the pairwise order model with any tapering coefficients c_h, $h = 1, \ldots, m-1$, the matrix R_π in assumption C is an orthogonal matrix.

Theorem 2 implies that under the pairwise order model with any tapering coefficients c_h, a design measure w is A- and MS-optimal if and only if $M(w) = M_0$, and is E-optimal if $M(w) = M_0$.

Although the proofs of optimality Theorem 1 amd 2 do not require explicit knowledge of M_0, we need to find M_0 and its eigenvalues to assess the efficiencies of a given design measure or exact design under, say, the D-criterion. M_0 and its eigenvalues can be evaluated via computation for small m, but this is rather unrealistic for

larger m, as computing $M_0 = (1/m!)\sum_{a\in S_m} x(a)x(a)'$ requires summing $m!$ matrices, of which the complexity grows superexponentially as m increases. This section is thus devoted to deriving explicit formulas for M_0 and its eigenvalues. Some more notations will help.

For the pairwise order model, recall that $S = \{ij : 1 \le i < j \le m\}$. Write I for the identity matrix of order $q = p - 1 = m(m-1)/2$. Define V as the $q \times q$ matrix, with rows and columns indexed by the elements of S, such that for $ij, kl \in S$, the (ij, kl)th element of V is

$$V(ij,kl) = \begin{cases} 1, & \text{if } i = k, j \neq l \text{ or } i \neq k, j = l, \\ -1, & \text{if } i = l \text{ or } j = k, \\ 0, & \text{otherwise.} \end{cases} \tag{3.1}$$

For instance, if $m = 4$, then $S = 12, 13, 14, 23, 24, 34$ and

$$V = \begin{bmatrix} 0 & 1 & 1 & -1 & -1 & 0 \\ 1 & 0 & 1 & 1 & 0 & -1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ -1 & 1 & 0 & 0 & 1 & -1 \\ -1 & 0 & 1 & 1 & 0 & 1 \\ 0 & -1 & 1 & -1 & 1 & 0 \end{bmatrix},$$

where $V(12,12) = 0$, $V(13,23) = 1$, $V(23,12) = -1$, and so on. Let

$$b_0 = 2\big\{(m-1)c_1^2 + \cdots + c_{m-1}^2\big\}\big/\big\{m(m-1)\big\}, \tag{3.2}$$

$$b_1 = 2\sum_h \big\{m - h(1) - h(2)\big\}c_{h(1)}\{2c_{h(1)+h(2)} - c_{h(2)}\}\big/\big\{m(m-1)(m-2)\big\}, \tag{3.3}$$

where $\sum_h$ denotes sum over positive integers $h(1)$, $h(2)$ such that $h(1)+h(2) \le m-1$. In general, (3.2) and (3.3) do not permit further simplification though, when $c_h = 1$ for all h, these reduce to

$$b_0 = 1,\ b_1 = 1/3. \tag{3.4}$$

Theorem 4. *The moment matrix of the uniform design measure w_0 is $M_0 = diag(b_0 I + b_1 V)$, having eigenvalues 1, $b_0 + (m-2)b_1$ and $b_0 - 2b_1$, with multiplicities 1, $m-1$ and $(m-1)(m-2)/2$, respectively.*

If the -1's in V were 1's, V would equal to an association matrix of the triangular association scheme (Raghavarao 1971, Chapter 8) and the related association algebra could be useful in studying the eigenvalues of M_0 in Theorem 2. Somewhat in the same spirit, the proof in the appendix obtains an expression for V^2 as a linear combination of I and V, and makes use of it.

Theorem 2 and 3 allow us to define the efficiencies of any design measure w, under the pairwise-order model and various optimality criteria, relative to the uniform hence optimal design measure w_0. In particular, the D- and A-effiiciencies of w, i.e., $\left[\det\{M(w)\} \big/ \det(M_0)\right]^{1/p}$ and $\operatorname{tr}\left(M_0^{-1}\right) \big/ \operatorname{tr}\{M^{-1}(w)\}$, are

$$\text{D-eff}(w) = \left[\frac{\det\left\{M(w)\right\}}{\left\{b_0 + (m-2)b_1\right\}^{m-1}(b_0 - 2b_1)^{(m-1)(m-2)/2}}\right]^{1/p}, \tag{3.5}$$

and

$$\text{A-eff}(w) = \frac{1 + (m-1)\left[\left\{b_0 + (m-2)b_1\right\}^{-1} + \left\{(m-2)/2\right\}(b_0 - 2b_1)^{-1}\right]}{\operatorname{tr}\left\{M^{-1}(w)\right\}}. \tag{3.6}$$

respectively. The design measure w in (3.5) and (3.6) can well correspond to an exact design.

Remark 2. Equations (3.5) and (3.6) are pivotal in efficiency calculation for a given choice of the c_h in (2.4) as well as studying robustness across such choices. For illustration, we study the performance of the order-of-addition orthogonal arrays of Voelkel (2019) under tapered models given by (i) $c_h = 1/h$ and (ii) $c_h = (1/2)^{h-1}$, $h = 1, \ldots, m-1$. By (3.5) and (3.6), for $m = 5$, the 12-run array in Voelkel (2019)'s Table 3 has D- and A-efficiencies 0.985 and 0.972 under (i), and 0.989 and 0.980 under (ii). These arrays were found earlier to be optimal under the original pairwise order model. Similarly, for $m = 6$, the 24-run arrays in his Table 7 are also quite robust, the best of these being the leftmost one in that table, having D- and A-efficiencies 0.992 and 0.984 under (i), and 0.994 and 0.988 under (ii). Indeed, even these figures can be conservative as, under (i) and (ii), there may not exist any exact design of the same run size as these arrays and having moment matrix M_0.

3.2. Optimal and robust fractional pairwise order designs

The literature on order-of-addition design lacks any systematic procedure for the construction of optimal fractional designs. To this end, we now propose a method that yields optimal fractions much smaller than the full design d_0 and has potential for further improvement, as discussed later in § 6. Our construction is based on the fact, noted in § 3, that an N-run exact design d, having the same moment matrix M_0 as d_0, is optimal among all N-run designs, for every criterion $\phi(\cdot)$ as in Theorem 1. Throughout this section, we work under the original pairwise order model as given by $c_h = 1$ for all h. Then by (3.4) and Theorem 2,

$$M_0 = \mathrm{diag}\{1, I + (1/3)V\}. \tag{3.7}$$

The resulting designs are seen to be very efficient under tapered models as well. We first illustrate the structure of the proposed designs, and then present the general construction procedure.

Let $m = 4$. Consider the 12-run half fractional design d, as given in transposed form by

$$\begin{array}{cccccccccccc} 1 & 2 & 4 & 3 & 1 & 3 & 4 & 2 & 1 & 4 & 3 & 2 \\ 2 & 1 & 3 & 4 & 3 & 1 & 2 & 4 & 4 & 1 & 2 & 3 \\ 3 & 4 & 1 & 2 & 2 & 4 & 1 & 3 & 2 & 3 & 1 & 4 \\ 4 & 3 & 2 & 1 & 4 & 2 & 3 & 1 & 3 & 2 & 4 & 1 \end{array}.$$

This can be obtained from the first design in Table 2 of Voelkel (arXiv:1701.02786v2) by relabeling of components and row permutation. The design d has moment matrix M_0 and hence enjoys the optimality properties of d_0. To see the structure of d, let

$$B_1 = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}, \overline{B}_1 = \begin{bmatrix} 3 & 4 \\ 4 & 3 \end{bmatrix}, B_2 = \begin{bmatrix} 1 & 3 \\ 3 & 1 \end{bmatrix}, \overline{B}_2 = \begin{bmatrix} 2 & 4 \\ 4 & 2 \end{bmatrix}, B_3 = \begin{bmatrix} 1 & 4 \\ 4 & 1 \end{bmatrix}, \overline{B}_3 = \begin{bmatrix} 2 & 3 \\ 3 & 2 \end{bmatrix}.$$

Then d can be expressed as $[D_1^\intercal, D_2^\intercal, D_3^\intercal]^\intercal$, where

$$D_u = \begin{bmatrix} B_u & \overline{B}_u \\ {\sim}\overline{B}_u & B_u \end{bmatrix}, \quad u = 1, 2, 3, \tag{3.8}$$

with $\sim$ representing the operator of column reversal of any matrix; for example, ${\sim}\overline{B}_1$ has first column $(4, 3)^\intercal$ and second column $(3, 4)^\intercal$.

We now show how the above design structure allows an extension to the case of general even $m \geq 4$. Let $s = m/2$, $L = m!/\{2(s!s!)\}$, and $\Gamma = \{1, \ldots, m\}$. Consider lexicographically arranged distinct sets $C_1, \ldots, C_L$, where each C_u consists of 1 and $s - 1$ other elements of Γ. For each u, let $\overline{C}_u$ be the complement of C_u in Γ. In

both C_u and $\overline{C}_u$, the elements are arranged in the ascending order. For $u = 1, \ldots, L$, let B_u be the $s! \times s$ array with rows formed by all permutations of the elements of C_u, and define $\overline{B}_u$ similarly with reference to $\overline{C}_u$. Thus, if $m = 4$, then $C_1 = \{1, 2\}$, $\overline{C}_1 = \{3, 4\}$, $C_2 = \{1, 3\}$, $\overline{C}_2 = \{2, 4\}$, $C_3 = \{1, 4\}$, $\overline{C}_3 = \{2, 3\}$, entailing B_u and $\overline{B}_u$ ($u = 1, 2, 3$) as shown above. Similarly, if $m = 6$, then for example, $C_5 = \{1, 3, 4\}$ and $\overline{C}_5 = \{2, 5, 6\}$, and hence

$$B_5 = \begin{bmatrix} 1 & 1 & 3 & 3 & 4 & 4 \\ 3 & 4 & 1 & 4 & 1 & 3 \\ 4 & 3 & 4 & 1 & 3 & 1 \end{bmatrix}^\top, \quad \overline{B}_5 = \begin{bmatrix} 2 & 2 & 5 & 5 & 6 & 6 \\ 5 & 6 & 2 & 6 & 2 & 5 \\ 6 & 5 & 6 & 2 & 5 & 2 \end{bmatrix}^\top.$$

For $u = 1, \ldots, L$, now define D_u as in (3.8). Along the lines of the 12-run design shown above for $m = 4$, let d^* be the design consisting of the $m!/s!$ treatments given by the rows of $D = [D_1^\top, \ldots, D_L^\top]^\top$.

Theorem 5. *For every even $m \geq 4$, the design d^* has moment matrix M_0 and is hence ϕ-optimal, among designs with the same number of runs, for every optimality criterion $\phi(\cdot)$ which is concave and signed permutation invariant.*

The above construction readily yields ϕ-optimal designs for odd $m = 2s+1$ ($s \geq 2$) as well. To get such a design in $(2s+1)!/s!$ runs, one has to stack $2s+1$ copies of D. Then it suffices to insert a column consisting only of $2s+1$ just before the lth column of the lth copy ($l = 1, \ldots, 2s$), and also after the last column of the last copy.

In order to explore the performance of the above fractional designs under tapered models, we again consider (i) $c_h = 1/h$, and (ii) $c_h = (1/2)^{h-1}$ ($h = 1, \ldots, m-1$). Quite reassuringly, the D- and A-efficiencies of our fractional designs for $m = 4, \ldots, 10$, calculated from (3.5) and (3.6), turn out to be over 0.99, under both (i) and (ii). For the reason indicated at the end of § 4, even these figures can be conservative.

Moreover, the same pattern is seen to persist for other choices of the c_h as well. This robustness facilitates the use of our fractional designs in model selection, allowing a wide range of models to choose from. One does not need to pre-specify the c_h in (2.4), but can make a post-experimentation data driven choice of these quantities with the assurance that the design will remain highly efficient under the model so reached. For illustration, let $m = 4$ and suppose the design d^* in Theorem 3 leads to observations

$$8.27, 3.09, 10.21, 7.26, 0.76, 12.15, 3.90, 5.09, 11.95, 1.62, 4.03, 7.77,$$

which correspond to treatments in the same order as in d^*, and are generated by adding perturbations, each uniform over $[-1, 1]$, to a random permutation of $1, \ldots, 12$. Standard cross-validation techniques may now be used to compare various choices of the c_h in (2.4). For instance, one can employ leave-one-out cross-validation and find the ratio of the predictive residual sum of squares for any choice of the c_h to that for a model involving only the general mean. A choice of the c_h may be considered satisfactory if this ratio is sufficiently small, say 0.1 or less. With observations as above, this ratio is found to be 0.083 for $c_h = 1/h$ $(h = 1, 2, 3)$. Thus, a model given by this choice of the c_h in (2.4) is reasonable and one remains assured of high efficiency of our fractional design under the model so reached. Indeed, as indicated in the next section, our designs remain highly efficient when the model incorporates some three-way orderings in addition to the pairwise terms, and thus allow even more flexibility in model selection, if necessary.

In the construction of optimal design d^* of Theorem 5, the rows of each B_u and $\overline{B}_u$ are arranged in a lexicographic model for convenience. In fact, if we randomly permute the rows of B_u, the resulting d^* remains optimal under the original pairwise-

order model. This produces a vast amount of non-isomorphic optimal designs, from which we can choose more robust designs. The design obtained via row permutations turn out to be, typically, highly robust to all the models given in Section 2.3. For example, when $m = 6$, an obtained design attains a D-efficiency of 0.912 under Mee (2019)'s triplet order model, 0.967 under Yang et al. (2019)'s component-position model, 0.998/0.975 under the first/second order model on simple orders, and 1.000/0.994 under the first/second order model on simple positions. When $m = 8$, an obtained design achieves a D-efficiency of 0.986 under the triplet order model, 0.995 under the component-position model, 1.000/0.996 under the first/second order model on simple orders, and 1.000/0.999 under the first/second order model on simple positions.

The optimal design d^* permits a natural blocking, with treatments arising from each D_u constituting one block. In model (2.3) for the treatment mean $\tau(a)$, then β_0 has to be replaced by a block effect parameter depending on the block where any treatment a in d^* appears, and interest is focused on $\widetilde{\beta} = (\beta_{12}, \ldots, \beta_{m-1\,m})^\top$. The proof of Lemma A4(a) in the Appendix shows that the blocking of d^* as envisaged above is a case of orthogonal blocking under the original pairwise order model. Therefore, under this model, the resulting block design remains ϕ-optimal for inference on $\widetilde{\beta}$, for every monotone criterion $\phi(\cdot)$ which is concave and signed permutation invariant, such as the D- and A-criteria. Moreover, for m up to 10, the D- and A-efficiencies of these blocked fractional designs, calculated from appropriate versions of (3.5) and (3.6), are again found to be over 0.99, under both tapered models as given by (i) and (ii) of the last paragraph.

3.3. Proofs

Proof of Theorem 2

(a) Consider a design measure $w = \{w(a) : a \in A\}$. Because $\phi(\cdot)$ is concave,

$$\phi\big\{(1/m!)\sum_{\pi} M(\pi w)\big\} \geq (1/m!)\sum_{\pi}\phi\big\{M(\pi w)\big\}. \tag{3.9}$$

Now, $\{\pi a : \pi \in A\} = A$, for every fixed $a \in A$, so that by (2.15), $\sum_{\pi} x(\pi a)x(\pi a)^{\top} = \sum_{a} x(a)x(a)^{\top} = m!M_0$. Hence by (2.14),

$$(1/m!)\sum_{\pi} M(\pi w) = (1/m!)\sum_{\pi}\sum_{a} w(a)x(\pi a)x(\pi a)^{\top} = \sum_{a} w(a)M_0 = M_0. \tag{3.10}$$

Any π leads to a signed permutation matrix $R(\pi)$ such that

$$\begin{aligned} M(\pi w) &= \sum_{a} w(a)x(\pi a)x(\pi a)^{\top} \\ &= R(\pi)^{\top}\Big\{\sum_{a} w(a)x(a)x(a)^{\top}\Big\}R(\pi) = R(\pi)^{\top}M(w)R(\pi), \end{aligned}$$

and hence $\phi\{M(\pi w)\} = \phi\{M(w)\}$, because $\phi(\cdot)$ is signed permutation invariant. Consequently, in view of (A3), from (A2) we get $\phi(M_0) \geq \phi\{M(w)\}$, and the result follows.

Proof of Theorem 3

Lemma 4. *(a) For $ij \in S$, $\sum_a z_{ij}(a) = 0$ and $\sum_a z_{ij}^2(a) = (m!)b_0$.*

(b) For i, j, k such that $1 \leq i < j < k \leq m$, $\sum_a z_{ij}(a)z_{ik}(a) = (m!)b_1$, $\sum_a z_{ik}(a)z_{jk}(a) = (m!)b_1$, and $\sum_a z_{ij}(a)z_{jk}(a) = -(m!)b_1$.

(c) For $ij, kl \in S$, if the sets $\{i, j\}$ and $\{k, l\}$ are disjoint, then $\sum_a z_{ij}(a)z_{kl}(a) = 0$.

Proof. (a) This follows from (2.4) and (3.2), noting that among $z_{ij}(a)$, $a \in A$, each of c_h and $-c_h$ occurs with frequency $(m-2)!(m-h)$ $(h = 1, \ldots, m-1)$.

(b) By (2.4), considering the positions, in increasing order, occupied by some permutation of i, j and k in any treatment,

$$\sum_a z_{ij}(a) z_{ik}(a) =$$
$$2\{(m-3)!\} \sum_u \left\{ c_{u(2)-u(1)} c_{u(3)-u(1)} - c_{u(2)-u(1)} c_{u(3)-u(2)} + c_{u(3)-u(1)} c_{u(3)-u(2)} \right\},$$

where $\sum_u$ denotes sum over integers $u(1), u(2), u(3)$ satisfying $1 \le u(1) < u(2) < u(3) \le m$. Given any positive integers $h(1), h(2)$ such that $h(1) + h(2) \le m-1$, there are $m - h(1) - h(2)$ triplets $\{u(1), u(2), u(3)\}$ as above such that $u(2) - u(1) = h(1)$ and $u(3) - u(2) = h(2)$. Therefore,

$$\sum_a z_{ij}(a) z_{ik}(a) =$$
$$2\{(m-3)!\} \sum_h \left\{ m - h(1) - h(2) \right\} \left\{ c_{h(1)} c_{h(1)+h(2)} - c_{h(1)} c_{h(2)} + c_{h(2)} c_{h(1)+h(2)} \right\},$$

where $\sum_h$ is as in (3.3). Because

$$\sum_h \left\{ m - h(1) - h(2) \right\} c_{h(2)} c_{h(1)+h(2)} = \sum_h \left\{ m - h(1) - h(2) \right\} c_{h(1)} c_{h(1)+h(2)}$$

by symmetry, the first of the three identities in (b) is now evident from (3.3). The other two identities follow similarly.

(c) This is evident from (2.4), noting that the set of treatments A can be partitioned into disjoint pairs such that the two treatments within each pair have the positions of i and j interchanged and every other component in the same position. □

Lemma 5. *(a) $V^2 = 2(m-2)I + (m-4)V$, (b) V has eigenvalues $m-2$ and -2, with*

respective multiplicities $m-1$ *and* $(m-1)(m-2)/2$.

Proof. (a) For any $ij, kl \in S$, write ρ for the (ij,kl)th element of V^2. It suffices to show that ρ equals $2(m-2)$ if $ij = kl$, and $(m-4)V(ij,kl)$ otherwise. As V is symmetric, ρ equals the scalar product of the ijth and klth rows of V. Thus $\rho = \mu_{++} - \mu_{+-} - \mu_{-+} + \mu_{--}$, where μ_{+-} is the number of positions with the ijth row having 1 and the klth row having -1, and so on. Part (a) now follows from (3.1) via consideration of the cases (i)-(vi) below. In what follows, $\mu = (\mu_{++}, \mu_{+-}, \mu_{-+}, \mu_{--})$.

(i) If $ij = kl$, then $\mu = (m-i+j-3, 0, 0, m+i-j-1)$; $\rho = 2(m-2)$.

(ii) If $i = k, j \neq l$, then $i \leq m-2$, $\mu = (m-i-2, 0, 1, i-1)$ or $(m-i-2, 1, 0, i-1)$, according as $j < l$ or $j > l$, respectively; $\rho = m-4$.

(iii) If $i \neq k$, $j = l$, then $j \geq 3$, $\mu = (j-3, 1, 0, m-j)$ or $(j-3, 0, 1, m-j)$, according as $i < k$ or $i > k$, respectively; $\rho = m-4$.

(iv) If $i = l$, then $i \geq 2$, $\mu = (1, m-i-1, i-2, 0)$; $\rho = -(m-4)$.

(v) If $j = k$, then $j \leq m-1$, $\mu = (1, j-2, m-j-1, 0)$; $\rho = -(m-4)$.

(vi) If none of the five cases above arises, then $\{i,j\}$ and $\{k,l\}$ are disjoint sets. In this situation, μ equals $(1,1,1,1)$ if $i > l$ or $k > j$, $(2,2,0,0)$ if $i < k < l < j$, $(2,0,2,0)$ if $k < i < j < l$, and $(2,1,1,0)$ if $i < k < j < l$ or $k < i < l < j$. So, $\rho = 0$.

(b) By (a), any eigenvalue, λ, of V satisfies $\lambda^2 = 2(m-2) + (m-4)\lambda$, and hence equals $m-2$ or -2. As $\mathrm{tr}(V) = 0$, these eigenvalues have multiplicities $m-1$ and $(m-1)(m-2)/2$, respectively. □

We now complete the proof of Theorem 2. The expression for M_0 follows from (2.15), (3.1) and Lemma A2, recalling that $x(a) = [1, z(a)^\top]^\top$. Hence by Lemma A3, eigenvalues of M_0 are as stated.

Proof of Theorem 4

Let $X = [X_0, X_{12}, \ldots, X_{m-1\,m}]$ be the model matrix of d^*, arising from (2.3) and (2.4), with $c_h = 1$ for all h. Thus, X_0 is a column of ones and each X_{ij} corresponds to β_{ij} $(ij \in S)$. Write $N = (m!)/(s!)$ for the run size of d^*. Because d^* has moment matrix $(1/N)X^\top X$, the result follows from (3.1), (3.7) and Lemma A4 below, which is akin to Lemma A2 but has a different proof.

Lemma 6. *(a) For $ij \in S$, $X_0^\top X_{ij} = 0$ and $X_{ij}^\top X_{ij} = N$. Also, $X_0^\top X_0 = N$.*

(b) For i, j, k satisfying $1 \le i < j < k \le m$, $X_{ij}^\top X_{ik} = N/3$, $X_{ik}^\top X_{jk} = N/3$, and $X_{ij}^\top X_{jk} = -N/3$.

(c) For $ij, kl \in S$, if the sets $\{i, j\}$ and $\{k, l\}$ are disjoint, then $X_{ij}^\top X_{kl} = 0$.

Proof. (a) Clearly, $X_{ij}^\top X_{ij} = X_0^\top X_0 = N$ for $ij \in S$, as X has elements ± 1. Next, let $G_u(ij)$ be the contribution of D_u to $X_0^\top X_{ij}$. By (2.4) and (3.8), $G_u(ij) = 0$, irrespective of whether i is in C_u or $\overline{C}_u$, and j is in C_u or $\overline{C}_u$. Hence $X_0^\top X_{ij} = 0$ for any $ij \in S$.

(b) Write G_u for the row vector with elements $G_u(ij, ik)$, $G_u(ik, jk)$ and $G_u(ij, jk)$, where $G_u(ij, ik)$ is the contribution of D_u to $X_{ij}^\top X_{ik}$, and $G_u(ik, jk)$, $G_u(ij, jk)$ are similarly defined. Because the rows of B_u and $\overline{B}_u$ in (3.8) are formed by all permutations of the elements of C_u and $\overline{C}_u$, respectively, in view of (3.1) and (3.7), the following hold.

(i) $G_u = (2/3)(s!)(1, 1, -1)$, if either $i, j, k \in C_u$, or $i, j, k \in \overline{C}_u$;

(ii) $G_u = 2(s!)(1, 0, 0)$, if either $j, k \in C_u$, $i \in \overline{C}_u$, or $i \in C_u$, $j, k \in \overline{C}_u$;

(iii) $G_u = 2(s!)(0, 1, 0)$, if either $i, j \in C_u$, $k \in \overline{C}_u$, or $k \in C_u$, $i, j \in \overline{C}_u$;

(iv) $G_u = 2(s!)(0, 0, -1)$, if either $i, k \in C_u$, $j \in \overline{C}_u$, or $j \in C_u$, $i, k \in \overline{C}_u$.

Because the situation (i) corresponds to $(2s-3)!/\{s!(s-3)!\}$ choices of u, and each of (ii), (iii), (iv) corresponds to $(2s-3)!/\{(s-1)!(s-2)!\}$ choices of u, part (b) follows after a little algebra.

(c) Let $G_u(ij,kl)$ be the contribution of D_u to $X_{ij}^\top X_{kl}$. Then by (2.4) and (3.8),

(i) $G_u(ij,kl) = 2(s!)$, if either $i,k \in C_u$, $j,l \in \overline{C}_u$ or $j,l \in C_u$, $i,k \in \overline{C}_u$;

(ii) $G_u(ij,kl) = -2(s!)$, if either $i,l \in C_u$, $j,k \in \overline{C}_u$ or $j,k \in C_u$, $i,l \in \overline{C}_u$;

and $G_u(ij,kl) = 0$ in all other situations. Part (c) is now immediate, because each of (i) and (ii) corresponds to $(2s-4)!/\{(s-2)!(s-2)!\}$ choices of u. □

Chapter 4

Swap-r algorithm for the construction of efficient, large-m order-of-addition designs

4.1. Introduction

Chapter 3 has obtained a class of fractional order-of-addition designs, which are optimal under the pairwise-order model and meanwhile highly efficient under all other models in Section 3.2. Although they are much smaller than the full design, these optimal fractions are still hardly affordable when m is large; for example, 30240 runs are required when $m = 10$.

On the other hand, while a variety of order-of-addition designs have been constructed in the recent literature, useful designs for large m are still absent, as explained below. Voelkel (2019) obtained (nearly-)optimal designs with small number of runs, but with m up to 7. Yang et al. (2019) obtained highly efficient designs under both pairwise-order and component-position model, but with m up to 11 and

the number of runs n being a multiple of $m(m-1)$. Zhao et al. (2019) obtained minimal-point designs for any m, but the D-efficiencies of their designs fall below 0.4 when $m \geq 8$. Thus, existing order-of-addition designs have limitations in either the number of components or the number of runs, or fall short in the efficiency.

In this chapter, we propose a new algorithm called Swap-r to construct efficient order-of-addition designs with large m and small number of runs. The r in Swap-r means an appropriate neighborhood-size, as will be explained in Section 4.4. For example, we have obtained a design that has a D-efficiency of 0.960 under non-tapered pairwise-order model, with $m = 10$ components and only $n = 90$ runs. As another example, a 50-component, 2450-run design has been obtained, which has a D-efficiency of 0.904 under non-tapered pairwise-order model.

Such large-m order-of-addition designs are useful in many realistic problems. As a chemistry-related example, Henry et al. (1996) considered the comparison among 12! possible orderings of 12 different drugs. In operations research, the popular job-scheduling problem (see, e.g., Pinedo 2016), which aims to determine an order of processing different jobs so that the final cost is minimized, can be viewed as an order-of-addition problem. As modern job scheduling problems often involve a large number of jobs, large-m order-of-addition designs are needed for potential applications therein.

The rest of this chapter is organized as follows. Section 4.2 brings some notations. Sections 4.3 and 4.4 describe the proposed algorithm. To effectively convey the ideas behind Swap-r, we start by introducing a basic Swap-2 in Section 4.3.1, which is a realization of coordinate-exchange in the order-of-addition problem. Swap-r inherits the ideas of basic Swap-2, but incorporates a number of new features to help escaping local optima and to accelerate the computation, as will be illustrated in Section 4.3.2 and 4.4. For the implementation of Swap-r, parameters are systematically tuned in

Section 4.5. Efficient designs with large m are obtained then and their efficiencies are tabulated in Section 4.6. Finally, we conclude by adding that the methodologies in Swap-r are quite general and extendable.

4.2. Notations and problem formulation

To clearly present the algorithm, the notations used in this chapter are slightly different from those in Chapters 2 and 3. Basically, each vector is now represented as a lower-case letter in bold italic, such as $\boldsymbol{x}(\cdot)$, each matrix is represented as an upper-case letter in bold, such as $\mathbf{X}$, and scalars are all in italic. Different elements of a vector or matrix are distinguished by subscripts, for example, a_i represent the ith element of vector $\boldsymbol{a}$. Different objects of the same type are distinguished by the superscript $^{(\cdot)}$, for example, $\boldsymbol{a}^{(i)}$ denotes the ith run of design matrix. Values of an object in different iterations are distinguished by the superscript $^{\langle\cdot\rangle}$, for example, $\lambda^{\langle i\rangle}$ indicates the value of λ in the ith iteration.

Explicitly, some frequently used objects are denoted as below. Each order is now represented as a vector $\boldsymbol{a} = [a_1, a_2, \ldots, a_m]^\top$, where $a_1, a_2, \ldots, a_m$ form a permutation of $1, 2, \ldots, m$. $\boldsymbol{x}(\boldsymbol{a})$ denotes the model point of order $\boldsymbol{a}$, following equation (2.1), and p denotes the length of the vector $\boldsymbol{x}(\cdot)$. For example, under the pairwise-order model, $\boldsymbol{x}(\boldsymbol{a}) = [1, z_{ij}(\boldsymbol{a})]_{1\le i<j\le m}$ and $p = \binom{m}{2} + 1$.

An n-run order-of-addition design chooses n orders, denoted as $\boldsymbol{a}^{(1)}, \boldsymbol{a}^{(2)}, \ldots, \boldsymbol{a}^{(n)}$. Thus, the $n \times m$ matrix $\mathbf{D} = \left[\boldsymbol{a}^{(1)}, \boldsymbol{a}^{(2)}, \ldots, \boldsymbol{a}^{(n)}\right]^\top$ is called an order-of-addition design matrix. Under model (2.1), $\mathbf{D}$ corresponds to a $n \times p$ model matrix $\mathbf{X} = \left[\boldsymbol{x}(\boldsymbol{a}^{(1)}), \boldsymbol{x}(\boldsymbol{a}^{(2)}), \ldots, \boldsymbol{x}(\boldsymbol{a}^{(n)})\right]^\top$. As an example, when $m = 3$, a full design, which

tests all the 3! = 6 orders, has a design matrix $\mathbf{D} = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \\ 2 & 1 & 3 \\ 2 & 3 & 1 \\ 3 & 1 & 2 \\ 3 & 2 & 1 \end{bmatrix}$, and a model matrix $\mathbf{X} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 \\ 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & -1 \end{bmatrix}$ under the pairwise-order model.

Given pre-specified m and n, our goal is to find a $n \times m$ design matrix $\mathbf{D}$ such that the corresponding moment matrix $\mathbf{M} = \mathbf{X}^\top\mathbf{X}/n$ is optimized in terms of some design criterion. For simplicity, this chapter focuses on the construction of D-optimal designs. Thus our goal is maximizes the determinant $|\mathbf{M}|$, or equivalently, $|\mathbf{X}^\top\mathbf{X}|$. Throughout, denote $\mathbf{C} = \mathbf{X}^\top\mathbf{X}$.

As far as we see, the search of D-optimal order-of-addition designs is challenging in two aspects. First, the feasible region expands rapidly, since the number of possible orders ($m!$) grows exponentially or even faster as a function of the m. Second, there appears to be a large amount of local optima. Frequent local optima are expected, in view of the heavy restrictions on the model matrix $\mathbf{X}$. For example, under the pairwise-order model, each model point $\boldsymbol{x}(\boldsymbol{a})$ is restricted by the transitive property: for any $1 \le j < k < l \le m$ and any order $\boldsymbol{a}$, $z_{jk}(\boldsymbol{a}) = z_{kl}(\boldsymbol{a}) = 1$ implies that $z_{jl}(\boldsymbol{a})$ must be 1, and $z_{jk}(\boldsymbol{a}) = z_{kl}(\boldsymbol{a}) = -1$ enforces $z_{jl}(\boldsymbol{a})$ to be -1. The transitive property further leads to heavy restrictions in the structure of $\mathbf{X}^\top\mathbf{X}$ – hence, a substantial number of local optima in the target function $\left|\mathbf{X}^\top\mathbf{X}\right|$ are not surprising. In the following two sections, we develop algorithms to address these challenges in an efficient manner.

4.3. The Swap-two algorithm

4.3.1. The basic algorithm

We first propose an algorithm called "Swap-two" for constructing efficient order-of-addition designs with any m and any $n \geq p$. Basically, the algorithm starts with an arbitrary order-of-addition design $\mathbf{D}$ with a non-singular model matrix. Then at each step of update, the algorithm evaluates the change in D-efficiency when swapping a pair of adjacent components in a row of $\mathbf{D}$, and determines whether to accept the swap.

The techniques of coordinate-exchange and simulated annealing play a key role in the realization of Swap-two. By swapping only adjacent components in a run of $\mathbf{D}$, we alter a small number coordinates in the model matrix $\mathbf{X}$ at a time. In particular, under the pairwise-order model, each swap of adjacent components leads to the change of only one coordinate in the model matrix. To evaluate the change in D-efficiency from such swap, the update formulas in single-coordinate-exchange (Meyer and Nachtsheim 1995) are applicable. The coordinate-exchange algorithm has been widely applied in different large-scale design problems. The Swap-two algorithm inherits the benefits of coordinate-exchange, for example, (1) it eliminates the need of enumerating or storing all the candidate points, and (2) it substantially reduces the computational complexity at each step, compared with the traditional point-exchange algorithms such as Fedorov (1972).

The coordinate exchange accepts an exchange (i.e., a swap here) if and only if the D-efficiency increases from such a swap. This approach falls short in that the solution is likely to be trapped in a local optimum near the initial design, which is especially unfavorable in the heavily restricted problem of order-of-addition design. Kuhfeld and Tobias (2005) suggested to resolve this issue using simulated anneal-

ing (Kirkpatrick et al. 1983), which allows, with probability, some worse coordinate exchanges. Kuhfeld and Tobias (2005) commented that "combining simulated annealing with coordinate-exchange gives [their SAS macro] the ability to handle very complicated restrictions that, to our knowledge, cannot be handled any other way." Following their suggestion, we conduct the Swap-two algorithm under simulated annealing. The detailed procedure is as follows. The input of the algorithm is a model in form (1), a pre-specified design size (m, n), as well as three tuning parameters: (i) $T_0 > 0$ as the initial temperature in the simulated annealing, (ii) $\alpha \in (0, 1)$ as the cooling rate in the simulated annealing, (iii) $\gamma \in (0, 1]$ as the ratio of runs to update at each iteration. The usage of these parameters will be seen later. Throughout, write $\mathbf{C} = \mathbf{X}'\mathbf{X}$. The output is an order-of-addition design $\mathbf{D}$ that attains nearly-optimal $|\mathbf{C}|$.

Step-0 Choose an arbitrary $n \times m$ initial design $\mathbf{D}$ with a non-singular $\mathbf{C}$. Let $T = T_0$.

Step-1 The k-exchange strategy is adopted, which means that we only update a γ-fraction of runs instead of all the n runs in one iteration. A popular criterion to select points to update is

$$v_i = \boldsymbol{x}'\big(\boldsymbol{a}^{(i)}\big)\mathbf{C}^{-1}\boldsymbol{x}\big(\boldsymbol{a}^{(i)}\big), \tag{4.1}$$

where $\boldsymbol{a}^{(i)}$ is the ith run of $\mathbf{D}$. v_i equals the variance of prediction at order $\boldsymbol{a}_i$. The points with lowest v_i's are considered unfavorable (see, e.g., Johnson and Nachtsheim 1983). The set of least-critical indices is then defined as $\mathcal{K} = \{i_1, i_2, \ldots, i_k\} \subset \{1, 2, \ldots, n\}$, where $k = \lceil n \cdot \gamma \rceil$, and $i_1, i_2, \ldots, i_k$ are the indices resulting in the k smallest v_i's.

Step-2 Sequentially for every $i \in \mathcal{K}$: sequentially for j from 1 to $(m-1)$, decide whether to swap the jth and $(j+1)$th components of $\boldsymbol{a}^{(i)}$ via the "Decision $S_2(i,j;T)$" process below, where S_2 stands for "Swap-two".

Step-3 Iterate Steps-1 and 2 until a termination condition is reached. After each iteration, T is shrunk to αT. In our implementation, the iteration terminates if the relative increase of D-criterion in the last 3 iterations does not achieve 10^{-4}.

Step-4 Conduct Steps-0 through 3 for different initial designs.

Decision $S_2(i,j;T)$

The update formulas here are summarized from Meyer and Nachtsheim (1995). Write $\boldsymbol{a}^{(i)}$ as $\boldsymbol{a}$ for simplicity. Let

$$\mathbf{A} = \left[1 - v_i\right]\mathbf{C}^{-1} + \mathbf{C}^{-1}\boldsymbol{x}(\boldsymbol{a})\boldsymbol{x}'(\boldsymbol{a})\mathbf{C}^{-1}. \tag{4.2}$$

Denote $\boldsymbol{a}^*$ the vector obtained by swapping the jth and $(j+1)$th elements of $\boldsymbol{a}$. Reordering coordinates if necessary, partition $\boldsymbol{x}(\boldsymbol{a})$ as

$$\boldsymbol{x}(\boldsymbol{a}) = \left[\boldsymbol{x}_1'(\boldsymbol{a}), \boldsymbol{x}_2'(\boldsymbol{a})\right]', \tag{4.3}$$

where $\boldsymbol{x}_1(\boldsymbol{a})$ includes all the coordinate(s) in which $\boldsymbol{x}(\boldsymbol{a})$ differs from $\boldsymbol{x}(\boldsymbol{a}^*)$, and $\boldsymbol{x}_2(\boldsymbol{a})$ includes the remaining coordinates. Partition $\mathbf{A}$ as $\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}$ in correspondence with the partition of $\boldsymbol{x}(\boldsymbol{a})$. Then by swapping the jth and $(j+1)$th

elements of $\boldsymbol{a}$, $|\mathbf{C}|$ will be multiplied by $1 + \Delta(\boldsymbol{a}, \boldsymbol{a}^*)$, where

$$\Delta(\boldsymbol{a}, \boldsymbol{a}^*) = \boldsymbol{x}_1'(\boldsymbol{a}^*)\mathbf{A}_{11}\boldsymbol{x}_1(\boldsymbol{a}^*) - \boldsymbol{x}_1'(\boldsymbol{a})\mathbf{A}_{11}\boldsymbol{x}_1(\boldsymbol{a}) + 2\boldsymbol{x}_2'(\boldsymbol{a})\mathbf{A}_{21}\big[\boldsymbol{x}_1(\boldsymbol{a}^*) - \boldsymbol{x}_1(\boldsymbol{a})\big].$$

$\Delta(\boldsymbol{a}, \boldsymbol{a}^*)$ is called the multiplicative change in the determinant, or the delta function (Fedorov 1972). If each coordinate of $\boldsymbol{x}(\boldsymbol{a})$ is two-level (either -1 or $+1$), we have $\boldsymbol{x}_1(\boldsymbol{a}^*) = -\boldsymbol{x}_1(\boldsymbol{a})$, and therefore

$$\Delta(\boldsymbol{a}, \boldsymbol{a}^*) = -4\boldsymbol{x}_2'(\boldsymbol{a})\mathbf{A}_{21}\boldsymbol{x}_1(\boldsymbol{a}). \tag{4.4}$$

A swap here proposes to update the ith run of $\mathbf{D}$ as $\boldsymbol{a}^*$. The swap is surely accepted if $\Delta(\boldsymbol{a}, \boldsymbol{a}^*) \geq 0$, and is accepted with probability $\exp\big(\Delta(\boldsymbol{a}, \boldsymbol{a}^*)/T\big)$ if $\Delta(\boldsymbol{a}, \boldsymbol{a}^*) < 0$. Once a swap is accepted, $\mathbf{C}^{-1}$ shall be updated as

$$\mathbf{C}^{-1} - \mathbf{C}^{-1}\mathbf{F}_1\big(\mathbf{I}_2 + \mathbf{F}_2'\mathbf{C}^{-1}\mathbf{F}_1\big)^{-1}\mathbf{F}_2'\mathbf{C}^{-1}, \tag{4.5}$$

with $\mathbf{I}_2$ being the 2×2 identity matrix, $\mathbf{F}_1 = [\boldsymbol{x}(\boldsymbol{a}^*), -\boldsymbol{x}(\boldsymbol{a})]$ being $p \times 2$, and $\mathbf{F}_2 = [\boldsymbol{x}(\boldsymbol{a}^*), \boldsymbol{x}(\boldsymbol{a})]$ being $p \times 2$. Accordingly, the quantity v_i for any $i \in \mathcal{K}$ and the matrix $\mathbf{A}$ need to be recalculated for further attempts of swap.

Remark 3. The implementation of all the algorithms in this article requires efficiently partitioning $\boldsymbol{x}(\cdot)$ as in equation (4.3). We have established formulas to determine the indices of $\boldsymbol{x}_1(\cdot)$ in $\boldsymbol{x}(\cdot)$, thereby $\boldsymbol{x}(\cdot)$ can be partitioned. See Section 4.8.1.

4.3.2. Fast update formulas

Consider the computational efficiency of the update formulas as described above. The evaluation of the multiplicative change in determinant, i.e., the delta function $\Delta\big(\boldsymbol{a}, \boldsymbol{a}^*\big)$, is crucial in all exchange algorithms. As implied by Meyer and Nachtsheim (1995), computing $\Delta(\boldsymbol{a}, \boldsymbol{a}^*)$ by equation (4.4) requires only $O(p)$ operations, in contrast to the Fedorov's exchange algorithm which requires $O(p^2)$ operations. However, equation (4.4) is only efficient when the matrix $\mathbf{A}$ is known. Whenever an exchange is accepted, the computational complexity becomes $O(p^2)$ for the next attempt of swap. This is because the decision on the next swap requires recalculating the $p \times p$ matrix $\mathbf{A}$ in (4.4), which, in view of (4), further requires recalculating $\mathbf{C}^{-1}$ and $v_i = \boldsymbol{x}(\boldsymbol{a})'\mathbf{C}^{-1}\boldsymbol{x}(\boldsymbol{a})'$. The update of $\mathbf{C}^{-1}$ is in particular burdensome.

The original coordinate-exchange algorithm forces the D-efficiency to ascend at every step, and therefore the swaps will be accepted very occasionally after the first few iterations. As such, $\mathbf{C}^{-1}$ has to be updated only occasionally. However, under simulated annealing, considerably more swaps will occur, and $\mathbf{C}^{-1}$ has to be updated frequently. A different update formula is desired in order to avoid recalculation of $\mathbf{C}^{-1}$ at each accepted swap. We adopt the classical update formulas for adding or removing a design point, given in Plackett (1950) and exploited in Mitchell (1974). We follow the notations in Section 3.1 in the descriptive below. Given any matrix $\mathbf{X}$ with a non-singular $\mathbf{C} = \mathbf{X}'\mathbf{X}$ and any run $\boldsymbol{x}$ to be added onto $\mathbf{X}$, it holds that:

$$\big|\mathbf{C} + \boldsymbol{x}\boldsymbol{x}'\big| = \big|\mathbf{C}\big|(1 + \boldsymbol{x}'\mathbf{C}^{-1}\boldsymbol{x}), \tag{4.6}$$

$$\big(\mathbf{C} + \boldsymbol{x}\boldsymbol{x}'\big)^{-1} = \mathbf{C}^{-1} - \frac{\mathbf{C}^{-1}\boldsymbol{x}\boldsymbol{x}'\mathbf{C}^{-1}}{1 + \boldsymbol{x}'\mathbf{C}^{-1}\boldsymbol{x}}. \tag{4.7}$$

If $\boldsymbol{x}$ is any run to be removed from the design, all the plus/minus in equations (4.6)

and (4.7) are reversed. Exploiting these formulas, we modify the Step-2 in Section 3.1 as below.

Step-2 **(Fast Swap-two)** Fix any $i \in \mathcal{K}$, let $\boldsymbol{a}$ denote the current ith run. Let $\boldsymbol{u} = \mathbf{C}^{-1}\boldsymbol{x}(\boldsymbol{a})$, and $\lambda = 1 - \boldsymbol{x}'(\boldsymbol{a})\boldsymbol{u}$. Then for $\mathbf{W} = \mathbf{C} - \boldsymbol{x}(\boldsymbol{a})\boldsymbol{x}'(\boldsymbol{a})$, in view of equations (4.6) and (4.7), we have $\left|\mathbf{W}\right| = \lambda \cdot \left|\mathbf{C}\right|$ and $\mathbf{W}^{-1} = \mathbf{C}^{-1} + \boldsymbol{u}\boldsymbol{u}'/\lambda$. Compute and store the quantities $\left|\mathbf{W}\right|$ and $\mathbf{W}^{-1}$.

Let $\lambda^{\langle 0\rangle} = 1/\lambda$ and $\boldsymbol{u}^{\langle 0\rangle} = \boldsymbol{u}/\lambda$. For j from 1 to $m-1$, decide whether to swap the jth and $(j+1)$th components of $\boldsymbol{a}$ via the "Decision $FS_2(i,j;T)$" process below, where FS_2 means "fast Swap-two". At the end of this loop, update $\mathbf{C}^{-1}$ as

$$\mathbf{C}^{-1} = \mathbf{W}^{-1} - \boldsymbol{u}^{\langle m-1\rangle}\big(\boldsymbol{u}^{\langle m-1\rangle}\big)'/\lambda^{\langle m-1\rangle}. \tag{4.8}$$

Decision $FS_2(i,j;T)$

Still, write $\boldsymbol{a}$ as the current ith run. Partition $\boldsymbol{x}(\boldsymbol{a})$ as in equation (4.3). Let $\mathbf{W}^{-1} = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{bmatrix}$ and $\boldsymbol{u}^{\langle j-1\rangle} = \begin{bmatrix} \boldsymbol{u}_1^{\langle j-1\rangle} \\ \boldsymbol{u}_2^{\langle j-1\rangle} \end{bmatrix}$, in correspondence with the partition of $\boldsymbol{x}(\boldsymbol{a})$. Evaluate the delta function

$$\begin{aligned}\widetilde{\Delta}(\boldsymbol{a},\boldsymbol{a}^*) = {} & \boldsymbol{x}_1'(\boldsymbol{a}^*)\mathbf{B}_{11}\boldsymbol{x}_1(\boldsymbol{a}^*) - \boldsymbol{x}_1'(\boldsymbol{a})\mathbf{B}_{11}\boldsymbol{x}_1(\boldsymbol{a}) + \\ & 2\big(\boldsymbol{x}_1'(\boldsymbol{a}^*) - \boldsymbol{x}_1'(\boldsymbol{a})\big)\big(\boldsymbol{u}_1^{\langle j-1\rangle} - \mathbf{B}_{11}\boldsymbol{x}_1(\boldsymbol{a})\big).\end{aligned}$$

In the situation that $\boldsymbol{x}_1(\boldsymbol{a}^*) = -\boldsymbol{x}_1(\boldsymbol{a})$,

$$\widetilde{\Delta}(\boldsymbol{a},\boldsymbol{a}^*) = -4\boldsymbol{x}_1'(\boldsymbol{a})\big(\boldsymbol{u}_1^{\langle j-1\rangle} - \mathbf{B}_{11}\boldsymbol{x}_1(\boldsymbol{a})\big). \tag{4.9}$$

A swap here proposes to replace the current run $\boldsymbol{a}$ by $\boldsymbol{a}^*$. The swap is surely accepted

when $\widetilde{\Delta}(\boldsymbol{a}, \boldsymbol{a}^*) \geq 0$, and is accepted with probability $\exp\big(\widetilde{\Delta}(\boldsymbol{a}, \boldsymbol{a}^*)/T\big)$ for a negative delta function. If the swap is accepted, then let $\lambda^{\langle j \rangle} = \lambda^{\langle j-1 \rangle} + \widetilde{\Delta}(\boldsymbol{a}, \boldsymbol{a}^*)$ and

$$\boldsymbol{u}^{\langle j \rangle} = \boldsymbol{u}^{\langle j-1 \rangle} + \begin{bmatrix} \mathbf{B}_{11} \\ \mathbf{B}_{21} \end{bmatrix} \big(\boldsymbol{x}_1(\boldsymbol{a}^*) - \boldsymbol{x}_1(\boldsymbol{a}) \big). \tag{4.10}$$

If the swap is not accepted, then let $\lambda^{\langle j \rangle} = \lambda^{\langle j-1 \rangle}$ and $\boldsymbol{u}^{\langle j \rangle} = \boldsymbol{u}^{\langle j-1 \rangle}$, while $\mathbf{D}$ and $\mathbf{X}$ remain unchanged.

The proofs of the new update procedure, particularly in terms of equations (4.8), (4.9) and (4.10), are given in Appendix B. For the pairwise-order model, $\boldsymbol{x}_1(\boldsymbol{a})$ and $\mathbf{B}_{11}$ are both scalars. Thus whenever a swap is accepted, Decision $FS_2(i, j; T)$ requires only $(p + 2)$ multiplications, 2 in equation (4.9) and p in equation (4.10). This is in contrast to Decision $S_2(i, j; T)$ of Section 3.1 which requires more than $2p^2$ multiplications to update $\mathbf{C}^{-1}$, v_i, and $\mathbf{A}$. Hence if frequent swaps are anticipated, the update formulas here are faster than the original formulas in Meyer and Nachtsheim (1995).

The proposed fast update formulas assume that $\mathbf{W} = \mathbf{X}'_{-i}\mathbf{X}_{-i}$ is invertible, where $\mathbf{X}_{-i}$ is obtained by removing the ith run from $\mathbf{X}$. Thus $\mathbf{X}_{-i}$ must be full-ranked, and in particular, n must be greater than $p + 1$. In other words, the fast update formulas are not applicable to the construction of minimal-point designs.

4.4. Fast Swap-r algorithm for $r \geq 3$

4.4.1. Basic method

At each step, Swap-two attempts to swap a pair of adjacent components, and decides whether to accept the swap. This can be extended to a Swap-r algorithm for any

$3 \le r \le m$, which permutes r adjacent components at each step. Thus at each step, Swap-r evaluates a neighborhood of $(r!-1)$ candidate designs, chooses the best candidate, and then decides whether to accept the candidate by the rule of simulated annealing. Consider Swap-three for example. Suppose that the initial design is $\mathbf{D} = \begin{bmatrix} \boldsymbol{a}' \\ \overline{\mathbf{D}} \end{bmatrix}$ with $\boldsymbol{a} = [a_1, a_2, a_3, \ldots, a_m]'$. Let $\overline{\boldsymbol{a}} = [a_4, a_5, \ldots, a_m]'$. At the first step, Swap-three evaluates the D-criterion of $\mathbf{D}_1 = \begin{bmatrix} a_2, a_1, a_3, \overline{\boldsymbol{a}}' \\ \overline{\mathbf{D}} \end{bmatrix}$, $\mathbf{D}_2 = \begin{bmatrix} a_2, a_3, a_1, \overline{\boldsymbol{a}}' \\ \overline{\mathbf{D}} \end{bmatrix}$, $\mathbf{D}_3 = \begin{bmatrix} a_3, a_2, a_1, \overline{\boldsymbol{a}}' \\ \overline{\mathbf{D}} \end{bmatrix}$, $\mathbf{D}_4 = \begin{bmatrix} a_3, a_1, a_2, \overline{\boldsymbol{a}}' \\ \overline{\mathbf{D}} \end{bmatrix}$ and $\mathbf{D}_5 = \begin{bmatrix} a_1, a_3, a_2, \overline{\boldsymbol{a}}' \\ \overline{\mathbf{D}} \end{bmatrix}$. Let $\mathbf{D}^*$ be the optimal design among $\mathbf{D}_i$'s $(1 \le i \le 5)$. Swap-three then compares the D-criteria of $\mathbf{D}^*$ and $\mathbf{D}$, and decide whether to replace $\mathbf{D}$ with $\mathbf{D}^*$ by the simulated annealing rule.

Note that the above sequence of design matrices $(\mathbf{D}, \mathbf{D}_1, \ldots, \mathbf{D}_5)$ satisfies the minimal-change requirement: each matrix can be obtained from its immediate predecessor by swapping just a pair of adjacent components. Therefore, the D-criterion of each matrix in the sequence can be evaluated from its predecessor using the delta function for fast Swap-two, i.e., equation (4.9). In general, given any $r \ge 3$, the $r!$ permutations of $\{1, 2, \ldots, r\}$ can be rearranged to meet such minimal-change requirement, using the Johnson-Trotter algorithm (Johnson 1963). Thus the delta function in fast Swap-two can be applied for assessing the D-criterion of any candidate design in Swap-r. This substantially improves the computational efficiency of Swap-r.

4.4.2. Algorithm details

While the basic idea of Swap-r is simple, its implementation is not straightforward. For implementation, we need to determine what quantities to keep track of, and give

update formulas for all these quantities. The explicit procedure is given here. Still, we follow the basic algorithm in Section 3.1, but modify Step 2 as below:

Step-2 (Fast Swap-r) Fix any $i \in \mathcal{K}$, let $\boldsymbol{a}$ be the ith run of $\mathbf{D}$, $\boldsymbol{u} = \mathbf{C}^{-1}\boldsymbol{x}(\boldsymbol{a})$, and $\lambda = 1 - \boldsymbol{x}'(\boldsymbol{a})\boldsymbol{u}$. Then for $\mathbf{W} = \mathbf{C} - \boldsymbol{x}(\boldsymbol{a})\boldsymbol{x}'(\boldsymbol{a})$, in view of equations (4.6) and (4.7), we have $\left|\mathbf{W}\right| = \lambda \cdot \left|\mathbf{C}\right|$ and $\mathbf{W}^{-1} = \mathbf{C}^{-1} + \boldsymbol{u}\boldsymbol{u}'/\lambda$. Compute and store the quantities $\left|\mathbf{W}\right|$ and $\mathbf{W}^{-1}$. Let $\lambda^{\langle 0\rangle} = 1/\lambda$ and $\boldsymbol{u}^{\langle 0\rangle} = \boldsymbol{u}/\lambda$.

For j from 1 to $m-r+1$, decide whether to rearrange the jth up to $(j+r-1)$th components of the current ith run, via the "Decision $FS_r(i, j; T)$" process below. "FS_r" means "fast Swap-r". At the end of this loop, update $\mathbf{C}^{-1}$ by

$$\mathbf{C}^{-1} = \mathbf{W}^{-1} - \boldsymbol{u}^{\langle m-r+1\rangle}\left(\boldsymbol{u}^{\langle m-r+1\rangle}\right)'/\lambda^{\langle m-r+1\rangle}.$$

Decision $FS_r(i, j; T)$

1. The given quantities are: (i) the current ith run $\boldsymbol{a}$, (ii) $\mathbf{W}^{-1}$, (iii) $\lambda^{\langle j-1\rangle}$ and (iv) $\boldsymbol{u}^{\langle j-1\rangle}$. Write $a_j, a_{j+1}, \ldots, a_{j+r-1}$ as the jth, $(j+1)$th, ..., $(j+r-1)$th components of $\boldsymbol{a}$.

2. (**Minimal-change r-permutation**) Using the Johnson-Trotter algorithm, we can obtain a sequence of runs $\boldsymbol{a}^{\langle 0\rangle}, \boldsymbol{a}^{\langle 1\rangle}, \ldots, \boldsymbol{a}^{\langle r!-1\rangle}$, which form a minimal-change path that traverses the neighborhood obtained by permuting the jth up to $(j + r - 1)$th components of $\boldsymbol{a}$. Explicitly,

   (i) $\boldsymbol{a}^{\langle 0\rangle}$ equals the current run $\boldsymbol{a}$.

   (ii) Let $a_k^{\langle l\rangle}$ denotes the kth components of $\boldsymbol{a}^{\langle l\rangle}$. Then for any $0 \le l_1, l_2 \le r! - 1$ and any $k \notin \{j, j+1, \ldots, j+r-1\}$, $a_k^{\langle l_1\rangle}$ is the same as $a_k^{\langle l_2\rangle}$.

(iii) Let $\boldsymbol{b}^{\langle l\rangle} = \left[a_j^{\langle l\rangle}, a_{j+1}^{\langle l\rangle}, \ldots, a_{j+r-1}^{\langle l\rangle}\right]'$. Then $\boldsymbol{b}^{\langle 0\rangle}, \boldsymbol{b}^{\langle 1\rangle}, \ldots, \boldsymbol{b}^{\langle r!-1\rangle}$ form a full permutation. That is, each $\boldsymbol{b}^{\langle l\rangle}$ is a permutation of $a_j, a_{j+1}, \ldots, a_{j+r-1}$, and $\boldsymbol{b}^{\langle l\rangle}$'s are mutually different.

(iv) For any $1 \le l \le r! - 1$, $\boldsymbol{a}^{\langle l\rangle}$ is obtained from $\boldsymbol{a}^{\langle l-1\rangle}$ by swapping only a pair of adjacent components. Throughout, suppose that $\boldsymbol{a}^{\langle l\rangle}$ is obtained from $\boldsymbol{a}^{\langle l-1\rangle}$ by swapping its κ_lth and $(\kappa_l + 1)$th components. This index κ_l, given by the Johnson-Trotter algorithm, is recorded for future use.

Remark. The Johnson-Trotter algorithm need not be conducted at every step. In the very beginning of the program, we shall obtain the minimal-change r-permutation above for $\boldsymbol{a}^{\langle 0\rangle} = [1, 2, \ldots, m]'$ and $j = 1$. From this result, it is easy to obtain the desired $\boldsymbol{a}^{\langle l\rangle}$'s and κ_l's $(1 \le l \le r! - 1)$ for any other choice of $\boldsymbol{a}^{\langle 0\rangle}$ and j.

3. (**Update formulas**) Let $\lambda^{\langle j,0\rangle} = \lambda^{\langle j-1\rangle}$ and $\boldsymbol{u}^{\langle j,0\rangle} = \boldsymbol{u}^{\langle j-1\rangle}$. Then, sequentially for l from 1 to $r! - 1$, consider changing the ith run from $\boldsymbol{a}^{\langle l-1\rangle}$ to $\boldsymbol{a}^{\langle l\rangle}$. Suppose, for simplicity, that $x(\cdot)$ has two levels -1 and $+1$. Then $\boldsymbol{x}(\boldsymbol{a}^{\langle l\rangle})$ is obtained from $\boldsymbol{x}(\boldsymbol{a}^{\langle l-1\rangle})$ by switching the sign(s) of some coordinate(s). The indices for such "switched coordinates" are determined by the aforementioned κ_l. Reordering coordinates if necessary, partition

$$\boldsymbol{x}\big(\boldsymbol{a}^{\langle l\rangle}\big) = \Big[\boldsymbol{x}_1'\big(\boldsymbol{a}^{\langle l\rangle}\big), \boldsymbol{x}_2'\big(\boldsymbol{a}^{\langle l\rangle}\big)\Big]', \tag{4.11}$$

where $x_1'\big(\boldsymbol{a}^{\langle l\rangle}\big)$ includes all the switched coordinates. Partition $\mathbf{W}^{-1} = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{bmatrix}$, in correspondence with the partition of $\boldsymbol{x}\big(\boldsymbol{a}^{\langle l\rangle}\big)$.

When replacing $\boldsymbol{a}^{\langle l-1\rangle}$ with $\boldsymbol{a}^{\langle l\rangle}$, $|\mathbf{C}|$ is multiplied by $1+\Delta^{\langle l\rangle}$, where

$$\Delta^{\langle l\rangle} = -4\boldsymbol{x}_1'\big(\boldsymbol{a}^{\langle l-1\rangle}\big)\Big[\boldsymbol{u}_1^{\langle l-1\rangle} - \mathbf{B}_{11}\boldsymbol{x}_1\big(\boldsymbol{a}^{\langle l-1\rangle}\big)\Big]. \tag{4.12}$$

For the evaluation of $\Delta^{\langle k\rangle}$ $(k \geq l)$, let $\lambda^{\langle j,l\rangle} = \lambda^{\langle j,l-1\rangle} + \Delta^{\langle l\rangle}$ and $\boldsymbol{u}^{\langle j,l\rangle} = \boldsymbol{u}^{\langle j,l-1\rangle} - 2\begin{bmatrix}\mathbf{B}_{11}\\ \mathbf{B}_{12}\end{bmatrix}\boldsymbol{x}_1\big(\boldsymbol{a}^{\langle l-1\rangle}\big)$.

In the loop for l from 1 to $r!-1$, the quantities $\lambda^{\langle j,l\rangle}$ and $\boldsymbol{u}^{\langle j,l\rangle}$ are recorded for each l. Other quantities can be over-written after each iteration, so as to reduce the space complexity.

4. Note that for each $l \in \{1,\ldots,r!-1\}$, $\lambda^{\langle j,l\rangle} - \lambda^{\langle j,0\rangle} = \sum_{t=1}^{l}\Delta^{\langle t\rangle}$, which is the multiplicative change in $|\mathbf{C}|$ when replacing the ith run $\boldsymbol{a}$ by $\boldsymbol{a}^{\langle l\rangle}$. We then find $s = \arg\max_{1\leq l\leq r!-1}\lambda^{\langle j,l\rangle}$. A swap here proposes to update the ith run as $\boldsymbol{a}^{\langle s\rangle}$. Such update is surely accepted if $\lambda^{\langle j,s\rangle} - \lambda^{\langle j,0\rangle} \geq 0$, and is accepted with probability $\exp\Big[\big(\lambda^{\langle j,l\rangle} - \lambda^{\langle j,0\rangle}\big)/T\Big]$ otherwise.

   If the update is accepted, let $\lambda^{\langle j\rangle} = \lambda^{\langle j,s\rangle}$ and $\boldsymbol{u}^{\langle j\rangle} = \boldsymbol{u}^{\langle j,s\rangle}$, and replace the ith run of $\mathbf{D}$ by $\boldsymbol{a}^{\langle s\rangle}$. Otherwise, let $\lambda^{\langle j\rangle} = \lambda^{\langle j-1\rangle}$ and $\boldsymbol{u}^{\langle j\rangle} = \boldsymbol{u}^{\langle j-1\rangle}$, and keep the design matrix unchanged.

4.5. Choosing parameters

A most efficient implementation of Swap-r requires tuning parameters, including (i) the neighborhood size r, (ii) the initial temperature T_0, (iii) the cooling rate α and (iv) the proportion γ of critical rows. This section proposes a procedure of choosing optimal parameter values, and make practical recommendations under the pairwise-order model. Under other models, parameters can be tuned likewise.

An important consideration in choosing the parameters is the trade-off between the number of initial designs one can test and the amount of improvement one can gain (on average) for each initial design. For instance, given an initial design, the updated design from Swap-3 tends to have a higher efficiency than that from Swap-2, while on the other hand, Swap-3 requires more computational time, than Swap 2, to update each initial design.

In consideration of such trade-off, we evaluate the performance of an algorithm by measuring the maximum efficiency it obtains within a time limit. As an illustration, set $(m, n) = (10, 47)$ and set $(T_0, \alpha, \gamma) = (1, 0.9, 0.5)$. Run the Swap-2/Swap-3 algorithms, respectively, under a time limit of 200 seconds. It turns out that the Swap-2 has updated 1569 initial designs and obtained a maximum D-efficiency of 0.823, while Swap-3 has updated 884 initial designs and obtained a maximum D-efficiency of 0.846. Swap-3 is more favorable in view of this. For a statistical comparison between the performances of Swap-2 and Swap-3, ten replications has been conducted, where Swap-2/3 were run respectively for 200 seconds in each replication. Swap-2 has obtained a maximum D-efficiency of 0.819/0.820/0.825/0.813/0.811/0.818/0.819/0.830/0.813/0.821, while Swap-3 has obtained a maximum D-efficiency of of 0.850/0.851/0.838/0.842/0.842/0.848/0.839/0.845/0.848/0.840, in each replication. A t-test indicates that Swap-3 achieves a higher maximum D-efficiency, on average, than Swap-2. Thus, one prefers $r = 3$ rather than 2, when $(m, n) = (10, 47)$ and $(T_0, \alpha, \gamma) = (1, 0.9, 0.5)$. A thorough comparison between different choices of parameters can be conducted in a similar manner. Such comparison protocol well fits the practical needs.

We consider five choices of swap size r: 2,3,4,5,6; four choices of cooling rate α: 0.8, 0.85, 0.9, 0.95; and three choices of fraction of runs for k-exchange, γ: 0.25, 0.5, 0.75. The ranges of these parameters are determined by some initial experiments,

and as will be shown later, the optimal choice of each parameter falls within the selected range. The candidate values of simulated annealing starting temperature T_0 are, however, chosen according to the number of runs. Here, we consider four design sizes $(m, n) = (10, 47), (10, 90), (12, 68), (12, 132)$, with n taking either $p + 1$ or $2q$. Recall that $q = \binom{m}{2}$ and $p = q + 1$ under the pairwise-order model. We consider eleven choices of T_0: $0, 1, \ldots, 9, 10$ when $n = p + 1$; and another eleven choices of T_0: $0, 0.1, \ldots, 0.9, 1$ when $n = 2q$.

Two time limits, $t_{\max} = 200$ or 1000 seconds, are further considered. Thus, under each of the eight scenarios of $(m, n, t_{\max})$, we conduct a full factorial experiment which contains $5 \times 11 \times 4 \times 3 = 660$ combinations of (r, T_0, α, γ). For each combination, the Swap-r algorithm is executed under the corresponding parameters with ten replications, and the obtained maximum efficiency is recorded as the response for each replication.

Multiple comparison with the best (MCB) fits our purpose to determine the optimal choices of r, T_0, α and γ. For example, MCB for the parameter r provides simultaneous confidence intervals for $\mu_r - \max_{s \neq r} \mu_s$, where μ_r indicates the average response obtained when using Swap-r ($r, s \in \{2, 3, 4, 5, 6\}$). Any r is identified as "among the best" if and only if its corresponding interval contains 0. We apply the classical Hsu (1984)'s MCB as well as the robust MCB method of Nelson (1993) and Nelson and Matejcik (1995). The robust MCB relaxes the assumption of homoscedastic and uncorrelated responses in Hsu's method, while both methods assume normality. In our numerical study, the responses are believed to be uncorrelated since the random seeds have been varied across different cases and replications. The homoscedasticity and normality assumptions, on the other hand, may be violated. Thus, we apply the Box-Cox transformation on the responses. For example, under the scenario $(m, n, t_{\max}) = (10, 50, 200)$, the response is transformed to $\left(1 - d_{\max}\right)^{-3}$,

where $d_{\max}$ is the original response, i.e., the obtained maximum D-efficiency.

Under each scenario of $(m, n, t_{\max})$, the optimal choices of r, T_0, α and γ found by MCB are listed in Table 1. Compared with the robust MCB, Hsu's MCB generally produces more conservative confidence intervals and therefore identifies more levels of parameters as "among the best". In Table 1, the numbers in parentheses are the optimal levels identified by Hsu's MCB but not the robust MCB, while other numbers are identified as optimal by both methods. For example, when $(m, n, t_{\max}) = (10, 47, 200)$, Hsu's method did not identify significant differences between $T_0 = 1, 2, 3, 4$, or 5, while the robust method concludes that $T_0 = 2, 3, 4$ produce significantly higher efficiencies than other choices of T_0.

Table 4.1: Best choices of parameters obtained by MCB under each scenario

(a) When $m = 10$

	$n = p + 1 = 47$		$n = 2q = 90$	
Parameter	$t_{\max} = 200$	$t_{\max} = 1000$	$t_{\max} = 200$	$t_{\max} = 1000$
T_0	$(6), (7), (8), 9, 10$	$(6), (7), (8), 9, 10$	0.1	0.1
r	$4, 5$	$(4), 5$	$4, 5$	$4, 5$
α	0.95	0.95	0.85	0.85
γ	1	1	$0.75, 1$	$0.75, 1$

(b) When $m = 12$

	$n = p + 1 = 68$		$n = 2q = 132$	
Parameter	$t_{\max} = 200$	$t_{\max} = 1000$	$t_{\max} = 200$	$t_{\max} = 1000$
T_0	$(6), (7), (8), 9, 10$	$(6), (7), (8), 9, 10$	0.1	0.1
r	$(4), 5$	$5, (6)$	$4, 5$	$4, 5$
α	0.95	0.95	0.85	$0.85, 0.9$
γ	1	1	1	$(0.75), 1$

As indicated in Table 4.1, the optimal parameter setting is rather robust to the change of $t_{\max}$. Fixing $n = p+1$ or $n = 2q$, respectively, the optimal parameter setting is robust to different m's as well. Thus, we can make a generic recommendation as below. When $n = p + 1$, $(T_0, \alpha) = (10, 0.95)$ is recommended. When $n = 2q$,

$(T_0, \alpha) = (0.1, 0.85)$ is recommended. For n being either $p+1$ or $2q$, we recommend Swap-5 with $\gamma = 1$.

4.6. Obtained design efficiencies

The Swap-r algorithm has been implemented in `R`. For m up to 50 and n being either $p+1$ or $2q$, a D-efficient design with m components and n runs has been produced by Swap-r, under the pairwise-order model. Recall that $q = \binom{m}{2}$ and $p = q + 1$ under this model. For each obtained design, its D-efficiency relative to the optimal design is given in Table 4.2 below. In our program, each design was obtained within a time limit of 5000 seconds. The parameters (r, T_0, α, γ) were chosen following the recommendations in Section 5, for example, r is chosen as 5 for any $m \geq 5$. The D-efficiency is calculated based on the theory in Peng et al. (2019).

It can be seen that when $n = 2q$, the D-efficiency obtained by Swap-r is greater than 0.9 for any $m \leq 50$. Such designs are highly efficient. Even when n is as small as $p+1$, which is one more than the minimal size, the D-efficiency stays above 0.6 for any $m \leq 45$.

As a comparison, we evaluate the performance of the Fedorov (1972) exchange algorithm exploited by Voelkel (2019) for constructing D-efficient pairwise-order design. Following Voelkel (2019), the Federov exchange algorithm is implemented via the `optFederov` function in the `R` package `AlgDesign` (Wheeler 2014). Still, Federov exchange was run within a time limit of 5000 seconds, for each design size. Note that when m is greater than 6, Swap-r consistently outperforms Federov exchange, for n being either $p+1$ or $2q$. Of course, Federov exchange is no longer applicable for $m > 10$ because of the space complexity.

Yang et al. (2019) has given order-of-addition designs, which are elaborately

constructed from orthogonal arrays, with m up to 11. Their designs are optimal under their component-position model (except for $m = 6$ and 10) and quite efficient under the pairwise-order model. Compared to their designs, the designs obtained by Swap-r are, not surprisingly, more efficient under the pairwise-order model for each m, while less efficient under the component-position model. On the other hand, Swap-r is applicable to any number of runs n, while the method in Yang et al. (2019) requires n to be a multiple of $m(m-1) = 2q$.

Swap-r is most appealing for its ability of tackling large m. As far we know, no order-of-addition design with $m > 11$ has been given in the literature, while Swap-r produces efficient designs with m up to 50.

Table 4.2: D-efficiencies obtained by Swap-r, compared with alternative algorithms

(a) When $m \leq 10$

m	$n = p+1$		$n = 2q$	
	Swap-r	Fedorov	Swap-r	Fedorov
4	0.896	0.896	1	1
5	0.939	0.903	0.966	0.971
6	0.899	0.914	0.970	0.976
7	0.873	0.873	0.975	0.973
8	0.880	0.856	0.971	0.970
9	0.856	0.829	0.967	0.966
10	0.838	0.818	0.964	0.960

(b) When $m > 10$

m	$n = p+1$	$n = 2q$
11	0.826	0.961
12	0.808	0.957
13	0.800	0.955
14	0.785	0.953
15	0.779	0.950
16	0.768	0.948
17	0.758	0.946
18	0.748	0.944
19	0.740	0.943
20	0.732	0.941
25	0.699	0.933
30	0.671	0.926
35	0.648	0.920
40	0.629	0.914
45	0.609	0.909
50	0.592	0.904

4.7. Key features of the proposed algorithm: a summary

In this article, we propose the Swap-r algorithm for constructing D-efficient designs under any order-of-addition linear model, with flexible design sizes. The essential ideas and detailed update formulas of the algorithm have been illustrated. Using the proposed algorithm, efficient order-of-addition designs with small number of runs have been obtained under the pairwise-order model. The obtained designs are up to 50 components, while the current literature provide designs up to only 11 components.

In terms of methodology, the innovative features of the proposed algorithm are summarized below. First, the flexibility in the neighborhood size allows a trade-off between the efficacy and computational time of each update. Second, the minimum-change path, coupled with newly proposed sequential update formulas, considerably accelerates the search of each neighborhood. Third, simulated annealing has been adopted to help escaping local optima, and fast update formulas have been developed accordingly. Finally, a factorial computer experiment, coupled with multiple comparison with the best, has been conducted to tune the parameters. We shall emphasize that these features are critical to the performance of Swap-r. As an evidence, the basic Swap-2 (in Section 3.1) without simulated annealing, which does not incorporate above features, perform much worse than the Swap-r incorporating these features. For example, when $m = 30$ and $n = p + 1$, the basic Swap-2 without simulated annealing achieves a D-efficiency of 0.520, while Swap-r attains a D-efficiency of 0.671. As such, a substantial part of this paper is devoted to illustrating the realization of these features.

The features above can be naturally extended to the design construction in other

problems. As an example, for the construction of factorial designs, Swap-r can be extended to an r-coordinate-exchange algorithm, which bridges the coordinate-exchange and Federov's exchange algorithm. For factorial designs, the minimum-change path for searching a neighborhood can be obtained from the minimum-level-change run orders (see, e.g., Cheng et al. 1998 and Peng and Lin 2019b). The update formulas in Section 4 can thus be applied straightforwardly. Following the spirit of Kuhfeld and Tobias (2005), we believe that the extension of Swap-r is useful for the construction of large-scale factorial designs, especially in the presence of heavy restrictions.

4.8. Theoretical supports

This sections gives more mathematical details that justify some of our claims in the algorithm description. First, corresponding to Remark 3, we establish efficient indexing of the elements in $\boldsymbol{x}(\cdot)$, so that the partition in equation (4.3) is possible. Second, we just the core part in the proposed algorithm: update formulas on Δ. We show that equations (4.4) and (4.9) do correctly update the D-efficiency of the design after swapping.

4.8.1. Indexing of order indicators

The implementation of the proposed algorithms requires finding the indices of the pairwise-wise indicator $z_{jk}(\cdot)$ given any $1 \le j < k \le m$. Arranging the $z_{jk}(\cdot)$'s in a lexicographic order of jk, we have:

Proposition 1. *For any $1 \le j < k \le m$, z_{jk} is the $(j-1)m + k - j(j+1)/2$ th pairwise-order indicator.*

Proof.

$$
\begin{aligned}
&\#\{st : st \text{ precedes } jk \text{ in the lexicographic order}\} \\
=&\{st : s = j, j < t < k\} + \sum_{s=1}^{j-1} \{st : s < t \le m\} \\
=&(k - j - 1) + \sum_{s=1}^{s=j-1} (m - s) \\
=&(k - j - 1) + (m - 1 + m - j + 1)(j - 1)/2 \\
=&(j - 1)m + k - j(j + 1)/2 - 1.
\end{aligned}
$$

Thus the rank of jk is $(j-1)m + k - j(j+1)/2$. □

The indexing in the higher-order models of Mee (2019) is more complicated, but follows the same idea. Consider the Triplet-order model for simplicity. The indicators in this model include: $z_{jk}(\cdot)$'s $(1 \le j < k \le m)$ in a lexicographic order of jk, $z_{jk}(\cdot) \times z_{jl}(\cdot)$'s $(1 \le j < k < l \le m)$ in a lexicographic order of jkl, and $z_{jk}(\cdot) \times z_{kl}(\cdot)$'s in a lexicographic order of jkl. To determine the index of each indicator, it suffices to determine the rank of each jkl in the lexicographic order. We have:

Proposition 2. *For any $1 \le j < k < l \le m$, jkl is the $j^3/6 + (1/2 - m/2)j^2 + (m^2/2 - m + 1/3)j - k^2/2 + (m - 1/2)k + l - m/2 - m^2/2$ th element in the set $\{rst : 1 \le r < s < t \le m\}$, under the lexicographic order.*

4.8.2. Proof of update formulas on Δ

Given an $n \times q$ matrix $\mathbf{X}$ with each element being either -1 or 1, let $\boldsymbol{x}$ be the 1st row of $\mathbf{X}$. Suppose we would like to sequentially switch the signs of a set of coordinates in $\boldsymbol{x}$. For any $k \le 1$, let S_k be any non-empty subset of $\{1, 2, \ldots, q\}$. S_k's could be overlapping with each other. Denote $x_0 = \boldsymbol{x}$, and for any $k \ge 1$, let $\boldsymbol{x}_k$ be the row

obtained by switching the signs of the coordinates in $\boldsymbol{x}$, with coordinate-indices in sets S_1, S_2, ..., up to S_k sequentially.

Let $\mathbf{C} = \mathbf{X}'\mathbf{X}$, $\lambda_0 = 1/\big(1-\boldsymbol{x}'\mathbf{C}^{-1}\boldsymbol{x}\big)$ and $\boldsymbol{u}_0 = \lambda_0\mathbf{C}^{-1}\boldsymbol{x}$. For any set S, define a $q\times|S|$ matrix $\mathbf{P}_S$ as follows. Suppose $S = \{j_1, j_2, \ldots, j_s\}$ with $j_1 < \cdots < j_s$. For $1 \le l \le s$, let the (l, j_l)th element of $\mathbf{P}_S$ be 1, and let other elements of $\mathbf{P}_S$ be 0. By the definition of $\boldsymbol{x}_k$'s, it is not difficult to show that $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} - 2\mathbf{P}_{S_k}\mathbf{P}'_{S_k}\boldsymbol{x}_{k-1}$ for any $k \ge 1$.

Denote $\mathbf{W} = \mathbf{X}'\mathbf{X} - \boldsymbol{x}\boldsymbol{x}'$, that is, $\mathbf{W} = \mathbf{X}'_{-1}\mathbf{X}_{-1}$ where $\mathbf{X}_{-1}$ is obtained by removing the first row from $\mathbf{X}$. Suppose $\mathbf{W}$ is invertible. For any $k \ge 1$, iteratively define

$$\lambda_k = \lambda_{k-1} - 4\big(\mathbf{P}'_{S_k}\boldsymbol{x}_{k-1}\big)'\mathbf{P}'_{S_k}\boldsymbol{u}_{k-1} + 4\big(\mathbf{P}'_{S_k}\boldsymbol{x}_{k-1}\big)'\big(\mathbf{P}'_{S_k}\mathbf{W}^{-1}\mathbf{P}_{S_k}\big)\big(\mathbf{P}'_{S_k}\boldsymbol{x}_{k-1}\big), \tag{4.13}$$

and

$$\boldsymbol{u}_k = \boldsymbol{u}_{k-1} - 2\big(\mathbf{W}^{-1}\mathbf{P}_{S_k}\big)\big(\mathbf{P}'_{S_k}\boldsymbol{x}_{k-1}\big). \tag{4.14}$$

Proposition 3. *For any $k \ge 0$, we have $\big|\mathbf{W} + \boldsymbol{x}_k\boldsymbol{x}'_k\big|/|\mathbf{W}| = \lambda_k$, and that $\big(\mathbf{W} + \boldsymbol{x}_k\boldsymbol{x}'_k\big)^{-1} = \mathbf{W}^{-1} - \boldsymbol{u}_k\boldsymbol{u}'_k/\lambda_k$.*

The proof of Proposition 3 is based upon two basic formulas below. For any $k \ge 0$,

$$\big|\mathbf{W} + \boldsymbol{x}_k\boldsymbol{x}'_k\big| = \big|\mathbf{W}\big|(1 + \boldsymbol{x}'\mathbf{W}^{-1}\boldsymbol{x}), \tag{4.15}$$

$$\big(\mathbf{W} + \boldsymbol{x}_k\boldsymbol{x}'_k\big)^{-1} = \mathbf{W}^{-1} - \frac{\mathbf{W}^{-1}\boldsymbol{x}\boldsymbol{x}'\mathbf{W}^{-1}}{1 + \boldsymbol{x}'\mathbf{W}^{-1}\boldsymbol{x}}. \tag{4.16}$$

We first introduce a lemma and then give the complete proof of Proposition 3 following the lemma.

Lemma 7. *For any $k \ge 0$, $\boldsymbol{u}_k = \mathbf{W}^{-1}\boldsymbol{x}_k$.*

Proof of Lemma 7. By equation (4.16),

$$\begin{aligned}\boldsymbol{x}_0'\mathbf{C}^{-1}\boldsymbol{x}_0 &= \boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0 - \frac{\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0'}{1+\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0} \\ &= \boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0\left(1-\frac{\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0'}{1+\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0}\right) = \frac{1}{1+\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0},\end{aligned}$$

and then $\lambda_0 = 1/(1-\boldsymbol{x}_0'\mathbf{C}^{-1}\boldsymbol{x}_0) = 1+\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0$. In view of this and equation (4.16), $\lambda_0\mathbf{C}^{-1} = \mathbf{W}^{-1} + \mathbf{W}^{-1}\big(\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0\big) - \mathbf{W}^{-1}\boldsymbol{x}_0\boldsymbol{x}_0'\mathbf{W}^{-1}$. Thus

$$\begin{aligned}\boldsymbol{u}_0 = \lambda_0\mathbf{C}^{-1}\boldsymbol{x}_0 &= \mathbf{W}^{-1}\boldsymbol{x}_0 + \mathbf{W}^{-1}\big(\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0\big)\boldsymbol{x}_0 - \mathbf{W}^{-1}\boldsymbol{x}_0\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0 \\ &= \mathbf{W}^{-1}\boldsymbol{x}_0 + \mathbf{W}^{-1}\boldsymbol{x}_0\big(\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0\big) - \mathbf{W}^{-1}\boldsymbol{x}_0\boldsymbol{x}_0'\mathbf{W}^{-1}\boldsymbol{x}_0 = \mathbf{W}^{-1}\boldsymbol{x}_0.\end{aligned}$$

Since $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} - 2\mathbf{P}_{S_k}\mathbf{P}_S'\boldsymbol{x}_{k-1}$, in view of equation (4.14), $\boldsymbol{u}_k - \boldsymbol{u}_{k-1} = \mathbf{W}(\boldsymbol{x}_k - \boldsymbol{x}_{k-1})$ for any $k \geq 1$. Thus $\boldsymbol{u}_k = \mathbf{W}^{-1}\boldsymbol{x}_k$ for any $k \geq 0$. □

Proof of Proposition 3. Note that $\boldsymbol{x}_k = \Lambda_k\boldsymbol{x}_{k-1}$, thus

$$\begin{aligned}&\frac{\big|\mathbf{W}+\boldsymbol{x}_k\boldsymbol{x}_k'\big|}{|\mathbf{W}|} - \frac{\big|\mathbf{W}+\boldsymbol{x}_{k-1}\boldsymbol{x}_{k-1}'\big|}{|\mathbf{W}|} \\ =&\boldsymbol{x}_k'\mathbf{W}^{-1}\boldsymbol{x}_k - \boldsymbol{x}_k'\mathbf{W}^{-1}\boldsymbol{x}_k \\ =&\big[(\boldsymbol{I}-2\mathbf{P}_{S_k}\mathbf{P}_{S_k}')\boldsymbol{x}_{k-1}\big]'\mathbf{W}^{-1}(\boldsymbol{I}-2\mathbf{P}_{S_k}\mathbf{P}_{S_k}')\boldsymbol{x}_{k-1} - \boldsymbol{x}_{k-1}'\mathbf{W}^{-1}\boldsymbol{x}_{k-1} \\ =&-4\big(\mathbf{P}_S'\boldsymbol{x}_{k-1}\big)'\mathbf{P}_S'\mathbf{W}^{-1}\boldsymbol{x}_{k-1} + 4\big(\mathbf{P}_S'\boldsymbol{x}_{k-1}\big)'\big(\mathbf{P}_S'\mathbf{W}^{-1}\mathbf{P}_S\big)\big(\mathbf{P}_S'\boldsymbol{x}_{k-1}\big),\end{aligned}$$

which equals $-4\big(\mathbf{P}_S'\boldsymbol{x}_{k-1}\big)'\mathbf{P}_S'\boldsymbol{u}_{k-1} + 4\big(\mathbf{P}_S'\boldsymbol{x}_{k-1}\big)'\big(\mathbf{P}_S'\mathbf{W}^{-1}\mathbf{P}_S\big)\big(\mathbf{P}_S'\boldsymbol{x}_{k-1}\big)$ by Lemma 7. Then by equation (4.13), $\big|\mathbf{W}+\boldsymbol{x}_k\boldsymbol{x}_k'\big|/|\mathbf{W}| - \big|\mathbf{W}+\boldsymbol{x}_{k-1}\boldsymbol{x}_{k-1}'\big|/|\mathbf{W}| = \lambda_k - \lambda_{k-1}$. It is easy to show that $\big|\mathbf{W}+\boldsymbol{x}_0\boldsymbol{x}_0'\big|/|\mathbf{W}| = \lambda_0$, and therefore $\big|\mathbf{W}+\boldsymbol{x}_k\boldsymbol{x}_k'\big|/|\mathbf{W}| = \lambda_k$ for any $k \geq 0$.

It follows immediately that $\big(\mathbf{W}+\boldsymbol{x}_k\boldsymbol{x}_k'\big)^{-1} = \mathbf{W}^{-1} - \boldsymbol{u}_k\boldsymbol{u}_k'/\lambda_k$, in view of equation (4.15) and Lemma 7. □

Chapter 5

Optimal design for estimating both factor effects and error variance

5.1. Introduction

To provide extra degrees of freedom for estimating the error variance (σ^2), one traditional design strategy is to add center points. Under a first-order main-effect model with k experimental factors, suppose one is asked to conduct an n-run design where n is slightly greater than $k+1$. The conventional wisdom is to perform a $(k+1) \times k$ orthogonal main-effect design, e.g., a Plackett-Burman (Plackett and Burman 1946) design, and then add $(n-k-1)$ center points. In this chapter, we show that adding center points may not be a favorable plan for estimating σ^2. We show that to perform the $n \times k$ design as a whole, a better and in fact optimal plan is to choose k column from an n-run orthogonal main-effec design. Optimality theorems are derived and illustrative examples are given.

As a motivative example, consider a study involving seven factors $x_1, \ldots, x_7$, via a linear first-order model $y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_7 x_{i7} + \epsilon_i$, where ϵ_i's are

independent and identically distributed (i.i.d.) with mean 0 and variance σ^2. Here we have eight effect parameters $\beta_0, \beta_1, \ldots, \beta_7$, as well as the error variance σ^2 to estimate. For estimation of $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_7)^\top$, an eight-run experiment will be sufficient. Suppose we decide to add four additional runs for estimating σ^2, thus a total of 12 runs is called for.

How to design such an experiment in 12 runs? The conventional wisdom is to add four center points, to a 2^{7-4} fractional factorial design, or here we call it an 8×7 "H-design"; see Design A in Table 5.1(a). We propose an alternative plan: choose any 7 columns from a 12×11 H-design; see Design B in Table 5.1(b). Throughout, an m-run H-design refers to any $m \times (m-1)$ matrix $\mathbf{H}$ such that $[\mathbf{1}, \mathbf{H}]$ forms an Hadamard matrix. (Here m is any multiplier of 4; $\mathbf{1}$ is a column of +1s; the two levels in the Hadamard matrix are labeled as "+" and "–".) As will be shown, Design B is more preferable than Design A in terms of estimating σ^2. Explicitly, $\mathrm{Var}(\hat{\sigma}^2)$ is smaller under Design B, where $\hat{\sigma}^2$ is the estimate of σ^2.

Table 5.1: Designs A and B for 12 runs and 7 factors

(a) Design A

−	−	−	+	+	+	−
+	−	−	−	−	+	+
−	+	−	−	+	−	+
+	+	−	+	−	−	−
−	−	+	+	−	−	+
+	−	+	−	+	−	−
−	+	+	−	−	+	−
+	+	+	+	+	+	+
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

(b) Design B

−	−	−	−	−	−	−
+	−	+	−	−	−	+
+	+	−	+	−	−	−
−	+	+	−	+	−	−
+	−	+	+	−	+	−
+	+	−	+	+	−	+
+	+	+	−	+	+	−
−	+	+	+	−	+	+
−	−	+	+	+	−	+
−	−	−	+	+	+	−
+	−	−	−	+	+	+
−	+	−	−	−	+	+

The rest of the chapter is organized as follows. In Section 5.2, we present the explicit problem formulation, and establish the optimality of Design B in estimating

σ^2. Under various common distributions, theoretical values of $\mathrm{Var}(\hat{\sigma}^2)$ have been evaluated for both Designs A and B. It is shown that Design B achieves a substantially less dispersed $\hat{\sigma}^2$ than Design A. Section 5.3 presents the summary of a lengthy simulation study. The simulation results support our theory, and show that the significance tests achieve more reliable performance under Design B. Sections 5.4 and 5.5 are devoted to a slightly different scenario, where designs are restricted to be a (saturated) main-effect design supplemented by several follow-up runs. A theory of the optimal follow-up design is established, and a convenient method is proposed to construct (nearly-)optimal follow-up designs. All proofs are given last in Section 5.6.

5.2. Main results

The motivating example is a special case of the general setup below. Consider, in general, the linear model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{5.1}$$

where ϵ_i's are assumed to be i.i.d. with $\mathrm{E}(\epsilon_i) = 0$ and $\mathrm{Var}(\epsilon_i) = \sigma^2$; $\mathbf{X} = (\mathbf{1}, \mathbf{D})$ with $\mathbf{1}$ being the intercept column and $\mathbf{D}$ being an $n \times k$ design matrix ($n > k$). The goal here is to estimate both $\boldsymbol{\beta}$ and σ^2 in an efficient manner. Two types of designs above are considered.

Design A: $\mathbf{D} = \begin{bmatrix} \mathbf{H}_k \\ \mathbf{O} \end{bmatrix}$, where $\mathbf{H}_k$ is any $(k+1) \times k$ H-design and $\mathbf{O}$ is a $(n-k-1) \times k$ zero-matrix. Note that this design is conventionally used in industry.

Design B: $\mathbf{D}$ consists of any k distinct columns of any n-run H-design.

The estimators of $\boldsymbol{\beta}$ and σ^2 can be obtained by the classical least square methods (Seber and Lee, 2012), namely,

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}, \text{ and} \tag{5.2}$$

$$\hat{\sigma}^2 = \frac{1}{n-p}\mathbf{y}^\top(\mathbf{I}_n - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top)\mathbf{y}, \tag{5.3}$$

where $p = k+1$ and $\mathbf{I}_n$ denotes the $n \times n$ identity matrix. It is well known that under the aforementioned model assumptions, $\hat{\boldsymbol{\beta}}$ is always unbiased (under any design), and $\mathrm{Var}(\hat{\boldsymbol{\beta}}) = \sigma^2\left(\mathbf{X}^\top\mathbf{X}\right)^{-1}$ (Seber and Lee, 2012). Note that $\mathbf{X}^\top\mathbf{X}$ equals to $\begin{bmatrix} n & \mathbf{O} \\ \mathbf{O} & p\mathbf{I}_k \end{bmatrix}$ and $n\mathbf{I}_p$, for Designs A and B respectively. Thus Design B attains a slightly smaller $\mathrm{Var}(\hat{\beta}_j)$ for any $1 \le j \le k$. That is, Design B is (slightly) more efficient in estimating all main effects.

For the estimation of σ^2, it is known that $\hat{\sigma}^2$ follows $\chi^2_{n-p}\cdot\sigma^2/(n-p)$, if ϵ_i's follow a normal distribution (Anderson, 1958). Here we study how the choice of design will affect (i) the variance of $\hat{\sigma}^2$ and (ii) the covariance between $\hat{\sigma}^2$ and each $\hat{\beta}_j$, under various error distributions. The following result (Bai and Silverstein, 2010) is useful to our work. Recall that $p = k+1$, $\mathbf{I}_n$ is the $n \times n$ identity matrix, and $\mathbf{X} = [\mathbf{1}, \mathbf{D}]$ is the model matrix. For the $\hat{\sigma}^2$ given by equation (5.3), it is shown that:

Lemma 8 (Bai and Silverstein, 2010)**.**

$$\mathrm{Var}(\hat{\sigma}^2) = \sigma^4\left(\frac{2}{n-p} + \frac{\mathrm{E}(\epsilon_1^4)/\sigma^4 - 3}{(n-p)^2}\cdot\sum_{i=1}^{n} G_{ii}^2\right), \tag{5.4}$$

where $(G_{11}, G_{22}, \ldots, G_{nn})$ *are the diagonal elements of the matrix* $\mathbf{I}_n - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$.

The error distribution affects $\mathrm{Var}(\hat{\sigma}^2)$ through the quantity $\mathrm{E}(\epsilon_1^4)/\sigma^4 - 3$, the so-called excess kurtosis (EK) of ϵ_i's. For normal ϵ_i's, the excess kurtosis equals 0, and

thus $\mathrm{Var}(\hat{\sigma}^2) = 2\sigma^4/(n-p)$. For distributions with a positive EK, $\mathrm{Var}(\hat{\sigma}^2)$ is greater than $2\sigma^4/(n-p)$; while for negative-EK distributions, $\mathrm{Var}(\hat{\sigma}^2) < 2\sigma^4/(n-p)$.

The choice of design affects $\mathrm{Var}(\hat{\sigma}^2)$ through the quantity $\sum_{i=1}^n G_{ii}^2$. When the error distribution has EK>0, from equation (5.4), a minimum $\sum_{i=1}^n G_{ii}^2$ yields a minimum $\mathrm{Var}(\hat{\sigma}^2)$. Specifically, as $\sum_{i=1}^n G_{ii}^2$ increases by 1, $\mathrm{Var}(\hat{\sigma}^2)$ increases by the amount of $\sigma^4 \cdot \mathrm{EK}\ /(n-p)^2$. Note that EK has no upper bound and could be quite large (see Table 5.2 for examples). As such, $\sum_{i=1}^n G_{ii}^2$ plays a critical rule for the purpose of reducing $\mathrm{Var}(\hat{\sigma}^2)$. On the other hand, when EK<0, $\mathrm{Var}(\hat{\sigma}^2)$ is below $2\sigma^4/(n-p)^2$ (thus is reasonably small), and EK has a lower bound of -2. In such case, $\sum_{i=1}^n G_{ii}^2$ is not so critical. Since the error distribution is typically unknown in practice, we recommend, in general, a design with minimum $\sum_{i=1}^n G_{ii}^2$ – thus Design B. This is formulated in Theorem 1 below.

Theorem 6. *Define G_{ii} as the i-th diagonal element of $\mathbf{I}_n - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$.*

(i) For Design A, $G_{ii} = 1/p - 1/n$ $(1 \le i \le p)$, and $G_{ii} = 1 - 1/n$ $(p+1 \le i \le n)$. Therefore $\sum_{i=1}^n G_{ii}^2 = (n-p)(np-2p+1)\,/(np)$.

(i) For Design B, $G_{ii} = 1 - p/n$ $(1 \le i \le n)$. Therefore $\sum_{i=1}^n G_{ii}^2 = (n-p)^2/n$.

(iii) For any $n \times p$ matrix $\mathbf{X}$, $\sum_{i=1}^n G_{ii}^2 \ge (n-p)^2/n$.

The values of $\mathrm{Var}(\hat{\sigma}^2)$ under different scenarios can thus be theoretically derived, as shown in Table 5.2. These values provide intuitive comparisons between Designs A and B in the efficiency of estimating σ^2. Note that the values of $\sum_{i=1}^n G_{ii}^2$ in Theorem 1 do not depend on the choice of H-design in Designs A and B. Nor do the values in Table 5.2.

Table 5.2: Theoretical values of $\mathrm{Var}(\hat{\sigma^2})/\sigma^4$ for Design A / Design B under difference error distributions and design sizes

Distribution (after linear transformation*)	Excess kurtosis	Design size (n, k)		
		$(12, 7)$	$(20, 15)$	$(40, 31)$
Normal	0	0.5 / 0.5	0.5 / 0.5	0.25 / 0.25
Laplace	3	1.13 / 0.75**	1.18 / 0.65	0.607 / 0.325
Exponential	6	1.77 / 1	1.85 / 0.8	0.963 / 0.4
$\chi^2(3)$	4	1.34 / 0.83	1.40 / 0.7	0.725 / 0.35
$t(5)$	6	1.77 / 1	1.85 / 0.8	0.963 / 0.4
Log-normal$(0, 0.5^2)$	5.90	1.74 / 0.992	1.83 / 0.795	0.951 / 0.397
Log-normal$(0, 1)$	111	23.9 / 9.74	25.5 / 6.05	13.4 / 3.02
GEV with shape parameter as 0	2.4	1.01 / 0.7	1.04 / 0.62	0.535 / 0.31
Uniform	-1.2	0.247 / 0.4	0.229 / 0.44	0.107 / 0.22
Beta(2,2)	-0.857	0.319 / 0.429	0.306 / 0.457	0.148 / 0.229
$0.5N(0, 1) + 0.5N(0, 2^2)$	1.08	0.728 / 0.59	0.744 / 0.554	0.378 / 0.277
$0.6N(0, 1) + 0.2N(-1, 2^2) + 0.2N(1, 2^2)$	0.83	0.835 / 0.632	0.858 / 0.579	0.438 / 0.290
$0.5N(-1, 1) + 0.5N(1, 1)$	-0.5	0.395 / 0.458	0.387 / 0.475	0.191 / 0.238

*: Every error distribution is linearly transformed so that $E(\epsilon_i) = 0$. Note that the kurtosis does not change under linear transformation.

**: The first value is for Design A, and the second value is for Design B.

We next evaluate $\mathrm{Cov}(\hat{\sigma}^2, \hat{\beta}_j)$'s under different designs. As will be shown below, $\hat{\sigma}^2$ is uncorrelated with every $\hat{\beta}_j$ $(j \geq 1)$ under both Designs A and B.

Lemma 9.

$$\mathrm{Cov}(\hat{\boldsymbol{\beta}}, \hat{\sigma}^2) = \frac{\mathrm{E}\,\epsilon_1^3}{n-p}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top[G_{11}, G_{22}, \ldots, G_{pp}]^\top, \tag{5.5}$$

where $\mathrm{Cov}(\hat{\boldsymbol{\beta}}, \hat{\sigma}^2) = \left[\mathrm{Cov}(\hat{\sigma}^2, \hat{\beta}_0), \mathrm{Cov}(\hat{\sigma}^2, \hat{\beta}_1), \ldots, \mathrm{Cov}(\hat{\sigma}^2, \hat{\beta}_k)\right]^\top$*, and* G_{ii} *is the* i*-th diagonal element of* $\mathbf{I}_n - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$.

With the G_{ii}'s for Designs A and B given in Theorem 1, it is easy to verify that $\mathbf{X}^\top[G_{11}, G_{22}, \ldots, G_{pp}]^\top = [n-p, 0, \ldots, 0]^\top$ for both designs. On the other hand, $(\mathbf{X}^\top\mathbf{X})^{-1}$ equals $\begin{bmatrix} n^{-1} & \mathbf{O} \\ \mathbf{O} & p^{-1}\mathbf{I}_{p-1} \end{bmatrix}$ for Design A and $n^{-1}\mathbf{I}_p$ for Design B. It follows that $\mathrm{Cov}(\hat{\boldsymbol{\beta}}, \hat{\sigma^2}) = \mathrm{E}\,\epsilon_1^3\,[(n-p)/n, 0, \ldots, 0]^\top$ for both designs. As a result, the estimation of σ^2 is uncorrelated with all the main effects under either Designs A or B.

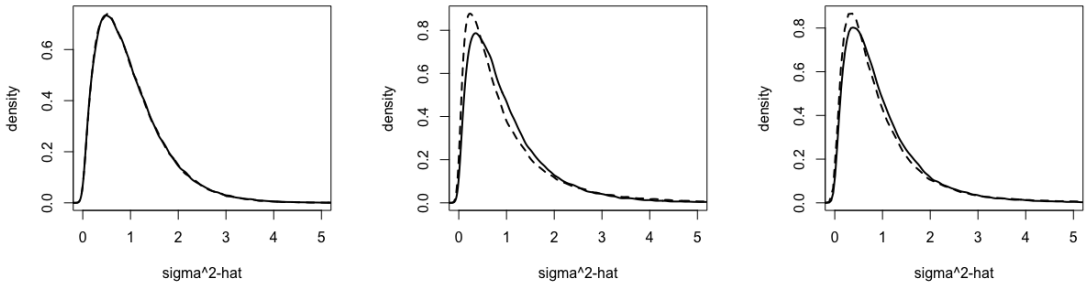
5.3. Simulation study

Some simulation results are provided here to demonstrate our theories in Section 5.2. Two examples with different design sizes are given. Section 5.3.1 studies the case with $(n, p) = (12, 8)$ and Section 5.3.2 studies $(n, p) = (40, 32)$. Under two common distributions: Laplace and $t(5)$, the numerical results support our conclusion in Section 5.2 that Design B provides more efficient estimates of σ^2 than Design A. In addition, it is found that Design B, as compared with Design A, achieves a more reliable type-I error and a higher power for the t- and F-tests.

5.3.1. Case-I: $(n, p) = (12, 8)$

Under different error distributions, we evaluate the sampling distributions of both Designs A and B (with 12 runs and 7 factors). The sampling distribution of $\hat{\sigma}^2$ is obtained via the following way. Set the true $\beta_0 = 2$, $\beta_j = 0.5$ $(1 \le j \le 7)$, and $\sigma^2 = 1$. Each time, draw a sample of ϵ_i's from the given distribution, generate y_i's under the given design, and then estimate σ^2 using Equation (5.3). Run this procedure for $100,000$ times. The sampling distributions of $\hat{\sigma}^2$ under the given design and error distribution can thus be obtained.

Figure 1 shows the kernel-smoothed densities (Wand and Jones, 1994) for the sampling distributions of $\hat{\sigma}^2$, under both Designs A and B and different error distributions. The result for each distribution is displayed in Figure 1, where the density for Design B is represented by a solid curve, and the density for Design A is represented by a dashed curve.



(a) Normal error (b) Laplace error (c) $t(5)$ error

Figure 5.1: Sampling distributions of $\hat{\sigma}^2$ under Designs A and B for different error distributions, when $(n, p) = (12, 8)$ (dashed curve: Design A; solid curve: Design B)

Figure 5.1 (a) shows the distributions of $\hat{\sigma}^2$ when each ϵ_i follows a standard normal distribution. As expected, the distributions of $\hat{\sigma}^2$ are the same under Designs A and B. Figure 5.1 (b) shows the distributions of $\hat{\sigma}^2$ when ϵ_i follows Laplace$(0, 1)/\sqrt{2}$ (the

Table 5.3: Quantiles of the sampling distribution of $\hat{\sigma}^2$, when $(n, p) = (12, 8)$

(a) When the error distribution is Laplace

Percentage	5%	10%	20%	50%	80%	90%	95%
Quantile under Design A	0.103	0.164	0.276	0.678	1.498	2.205	2.991
Quantile under Design B	0.142	0.216	0.346	0.761	1.510	2.072	2.660

(b) When the error distribution is $t(5)$

Percentage	5%	10%	20%	50%	80%	90%	95%
Quantile under Design A	0.129	0.197	0.315	0.692	1.419	2.042	2.805
Quantile under Design B	0.150	0.226	0.355	0.758	1.459	2.008	2.610

divisor $\sqrt{2}$ is for normalizing ϵ_i so that $\sigma^2 = 1$). It is evident $\hat{\sigma}^2$ is less dispersed under Design B, which agrees with Theorem 1. (Under Design A, $\hat{\sigma}^2$ is more skewed towards the right.) As another way of comparison, the quantiles of $\hat{\sigma}^2$ under Designs A and B are displayed in Table 5.3 (a). Almost all the quantiles are closer to the true value ($\sigma^2 = 1$) under Design B, which clearly indicates that Design B achieves a less dispersed estimate of error variance.

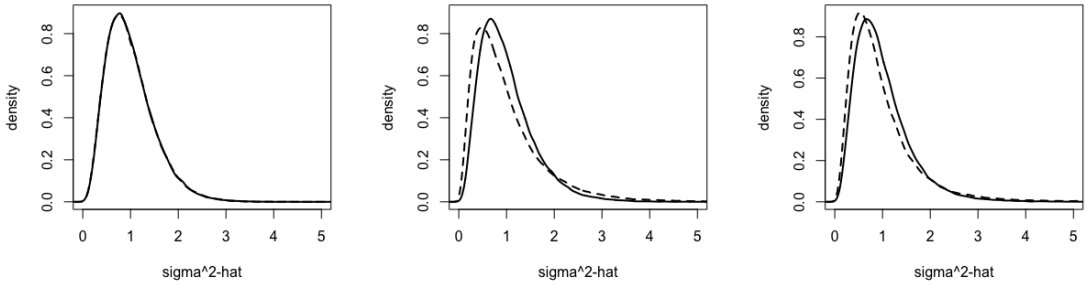
Figure 5.1 (c) and Table 5.3 (b) exhibit the distributions of $\hat{\sigma}^2$ when ϵ_i follows a $t(5)/\sqrt{3}$ distribution (the divisor $\sqrt{3}$ is chosen to guarantee $\sigma^2 = 1$). The pattern here is very similar to that under Laplace distribution. Both the density plots and quantiles indicate that $\hat{\sigma}^2$ is slightly less dispersed under Design B.

The difference in the sampling distributions of $\hat{\sigma}^2$ under Designs A and B generally becomes more evident when n and p increase, as will be shown below (Section 5.3.2).

5.3.2. Case-II: $(n, p) = (40, 32)$

Consider Designs A and B of a different size: $(n, p) = (40, 32)$. The sampling distributions of $\hat{\sigma}^2$ are demonstrated in Figure 5.2 and Table 5.4, which are analogies of those in Case I. We follow the simulation settings in Section 5.3.1: $100,000$ random samples were drawn to generate each plot and table. The true σ^2 is set as 1 and $\boldsymbol{\beta}$

is set as $(2, 0.5, \ldots, 0.5)$.



(a) Normal error (b) Laplace error (c) $t(5)$ error

Figure 5.2: Sampling distributions of $\hat{\sigma}^2$ under Designs A and B for different error distributions, when $(n, p) = (40, 32)$ (dashed curve: Design A; solid curve: Design B)

Table 5.4: Quantiles of the sampling distribution of $\hat{\sigma}^2$, when $(n, p) = (40, 32)$

(a) When the error distribution is Laplace

Percentage	5%	10%	20%	50%	80%	90%	95%
Quantile under Design A	0.212	0.290	0.414	0.794	1.450	1.960	2.480
Quantile under Design B	0.307	0.397	0.531	0.886	1.401	1.748	2.084

(b) When the error distribution is $t(5)$

Percentage	5%	10%	20%	50%	80%	90%	95%
Quantile under Design A	0.249	0.326	0.446	0.784	1.365	1.825	2.356
Quantile under Design B	0.306	0.397	0.528	0.876	1.383	1.740	2.096

The patterns here are very similar to those in Section 5.3.1 (with perhaps slightly more evidence). Under either Laplace or $t(5)$ errors, the sampling distribution of $\hat{\sigma}^2$ is apparently more concentrated towards the true value 1 under Design B than under Design A. (In fact, under Design B, the sampling distribution of $\hat{\sigma}^2$ deviates less than that under normality.)

In many situations, the reliability of significant tests, such as t- and F-tests, are of interest. Intuitively, a more efficient estimate of σ^2 will yield more reliable

test results. To confirm this, we next evaluate the powers of t- and F-test under both Designs A and B. Set the significance level at 0.05. Let β_j's $(1 \leq j \leq p-1)$ gradually deviate from 0 while fixing the intercept $\beta_0 = 2$. For each true $\boldsymbol{\beta}$, draw $100,000$ random samples of $\mathbf{y}$ from model (1) under both Designs A and B. Determine whether each sample of $\mathbf{y}$ falls in the critical regions of (i) the t-test for $H_0 : \beta_1 = 0$ and (ii) the F-test for $H_0 : \beta_j = 0$ (for all $1 \leq j \leq p-1$), and then estimate the powers of t- and F-tests for this particular $\boldsymbol{\beta}$. Simulations have been conducted, for $(n, p) = (12, 8)$ and $(40, 32)$ respectively, under different settings of $\boldsymbol{\beta}$. The results are summarized in Tables 5.5 and 5.6 below.

Table 5.5: Powers of t- and F- test under different $\boldsymbol{\beta}$'s for Designs A and B, when $(n, p) = (12, 8)$

(a) When the error distribution is $t(5)$

True $\boldsymbol{\beta}$	t-test for β_1		F-test	
	Under Design A	Under Design B	Under Design A	Under Design B
$\beta_j = 0$ $(1 \le j \le 31)$	0.0656	0.0457	0.0799	0.0468
$\beta_j = 0.8$ $(1 \le j \le 31)$	0.2423	0.2991	0.334	0.4108
$\beta_j = 1.6$ $(1 \le j \le 31)$	0.621	0.7609	0.7932	0.9016
$\beta_j = 2.4$ $(1 \le j \le 31)$	0.8824	0.9555	0.9574	0.9904
$\beta_j = 3.2$ $(1 \le j \le 31)$	0.9679	0.9916	0.9902	0.9987
$\beta_1 = 0.5$, $\beta_j = 0$ $(2 \le j \le 31)$	0.4665	0.5944	0.1664	0.1702
$\beta_1 = 1$, $\beta_j = 0$ $(2 \le j \le 31)$	0.9085	0.9676	0.4473	0.5556
$\beta_1 = 1.5$, $\beta_j = 0$ $(2 \le j \le 31)$	0.9874	0.9981	0.7283	0.8539
$\beta_1 = 2$, $\beta_j = 0$ $(2 \le j \le 31)$	0.9977	0.9998	0.8934	0.9627

(b) When the error distribution is Uniform

True $\boldsymbol{\beta}$	t-test for β_1		F-test	
	Under Design A	Under Design B	Under Design A	Under Design B
$\beta_j = 0$ $(1 \le j \le 31)$	0.0320	0.0542	0.0191	0.0554
$\beta_j = 0.5$ $(1 \le j \le 31)$	0.1543	0.2528	0.1763	0.3496
$\beta_j = 1$ $(1 \le j \le 31)$	0.5429	0.724	0.7797	0.9201
$\beta_j = 1.5$ $(1 \le j \le 31)$	0.898	0.9776	0.9977	0.9998
$\beta_j = 2$ $(1 \le j \le 31)$	0.9943	1	1	1
$\beta_1 = 0.8$, $\beta_j = 0$ $(2 \le j \le 31)$	0.3636	0.5324	0.064	0.1602
$\beta_1 = 1.6$, $\beta_j = 0$ $(2 \le j \le 31)$	0.9348	0.9908	0.2772	0.4876
$\beta_1 = 2.4$, $\beta_j = 0$ $(2 \le j \le 31)$	0.9998	1	0.6723	0.85
$\beta_1 = 3.2$, $\beta_j = 0$ $(2 \le j \le 31)$	1	1	0.9472	0.9879

Table 5.6: Powers of t- and F- test under different $\boldsymbol{\beta}$'s for Designs A and B, when $(n, p) = (40, 32)$

(a) When the error distribution is $t(5)$

True $\boldsymbol{\beta}$	t-test for β_1		F-test	
	Under Design A	Under Design B	Under Design A	Under Design B
$\beta_j = 0$ $(1 \le j \le 31)$	0.0784	0.0475	0.1333	0.0468
$\beta_j = 0.2$ $(1 \le j \le 31)$	0.2141	0.2114	0.4507	0.4518
$\beta_j = 0.4$ $(1 \le j \le 31)$	0.5461	0.62	0.8976	0.9733
$\beta_j = 0.6$ $(1 \le j \le 31)$	0.8362	0.907	0.992	0.9999
$\beta_j = 0.8$ $(1 \le j \le 31)$	0.9611	0.9876	0.9994	1
$\beta_1 = 0.4$, $\beta_j = 0$ $(2 \le j \le 31)$	0.5457	0.6206	0.1736	0.0832
$\beta_1 = 0.8$, $\beta_j = 0$ $(2 \le j \le 31)$	0.9628	0.9876	0.3001	0.2338
$\beta_1 = 1.2$, $\beta_j = 0$ $(2 \le j \le 31)$	0.9992	1	0.4961	0.5199
$\beta_1 = 1.6$, $\beta_j = 0$ $(2 \le j \le 31)$	1	1	0.6941	0.7912

(b) When the error distribution is Uniform

True $\boldsymbol{\beta}$	t-test for β_1		F-test	
	Under Design A	Under Design B	Under Design A	Under Design B
$\beta_j = 0$ $(1 \le j \le 31)$	0.0348	0.0501	0.0116	0.0512
$\beta_j = 0.2$ $(1 \le j \le 31)$	0.1439	0.196	0.242	0.4234
$\beta_j = 0.4$ $(1 \le j \le 31)$	0.4948	0.5968	0.9891	0.9854
$\beta_j = 0.6$ $(1 \le j \le 31)$	0.8529	0.9158	1	1
$\beta_j = 0.8$ $(1 \le j \le 31)$	0.9829	0.9944	1	1
$\beta_1 = 0.4$, $\beta_j = 0$ $(2 \le j \le 31)$	0.4948	0.5973	0.022	0.0841
$\beta_1 = 0.8$, $\beta_j = 0$ $(2 \le j \le 31)$	0.9827	0.9947	0.0858	0.2194
$\beta_1 = 1.2$, $\beta_j = 0$ $(2 \le j \le 31)$	1	1	0.3072	0.49
$\beta_1 = 1.6$, $\beta_j = 0$ $(2 \le j \le 31)$	1	1	0.6932	0.7886

These results clearly show that under $t(5)$ distribution (positive EK), the type-I error is always controlled by the significance level under Design B, while the type-I error considerably exceeds the significance level under Design A (especially for the F-test). Moreover, as long as $\boldsymbol{\beta}$ is not too small, Design B achieves a higher power than Design A. It is also shown that results under uniform error distribution (negative EK), the type-I error is well-controlled under both Designs A and Design B. On the other hand, Design B still achieves a higher power than Design A under uniform

errors. As pointed out by one referee of our work, the higher power under Design B is partially ascribed to its higher D-criterion. It will be explored in the future how the advantages of Design B in significance tests is connected to the fact that Design B attains the minimum $\sum_{i=1}^{n} G_{ii}^2$.

In summary, this section compares the sampling distributions of $\hat{\sigma}^2$ for Designs A and B, under different scenarios. The numerical results demonstrate that Design B achieves a less dispersed $\hat{\sigma}^2$; this supports the theory in Section 5.2. It is also shown that Design B achieves robust type-I errors (i.e., the type-I errors are closer to the significance level) and higher powers of t- and F-test, as compared to Design A.

5.4. The follow-up design problem: optimality theorems

As a sequel, we study the design problem following an initial main-effect design. Suppose an experimenter has first conducted a main-effect experiment with an $p \times k$ design, D_0 (recall that $p = k + 1$). It is well known that under the effect sparsity assumption (Box and Meyer 1986), a saturated main-effect design is sufficient for identifying the significant factors. Many inference methods are available under the effect sparsity, for example, Daniel (1959), Dong (1993), Lenth (1989), Ye et al. (2001), and Miller (2005). However, the effect sparsity assumption may not hold in some situations (see, e.g., Hurley 1995). In such cases, a follow-up experiment is needed to provide extra degrees of freedom for estimating error variance.

Our goal is to find an optimal $(n-p) \times k$ design D_a, so that the combined design $D = \begin{bmatrix} D_0 \\ D_a \end{bmatrix}$ achieves an efficient estimate of both $\boldsymbol{\beta}$ and σ^2. Again, D is slightly larger than D_0, i.e., n exceeds p by only a few runs. Assume that D_0 is a Hadamard matrix.

Also assume that each element of D_a is within the interval $[-1, 1]$. In Chapter 3.4 and 3.5, always denote $X = (\mathbf{1}, D)$, $X_0 = (\mathbf{1}, D_0)$, and $X_a = (\mathbf{1}, D_a)$, where $\mathbf{1}$ is the intercept column.

Remark 4. The conventional choice of D_a is $(n - p)$ runs of center points (so that Design A is the combined design). Design B is not considered in the follow-up design problem, because it does not include any $p \times k$ orthogonal sub-design.

In essence, we have three criteria for finding an optimal D_a: (i) maximize the estimation efficiency of $\hat{\boldsymbol{\beta}}$; (ii) maximize the estimation efficiency of $\hat{\sigma}^2$, which, based on the theory in Chapter 3.2, usually requires minimizing the quantity $\sum_{i=1}^{n} G_{ii}^2$; (iii) minimize the absolute correlation between $\hat{\sigma}^2$ and each β_j, $1 \le j \le k$. This section focuses on the optimality theory regarding these three criteria. (The next section will compare several typical types of designs in terms of these criteria, and a recommendation is made therein.)

First, for the estimation of $\boldsymbol{\beta}$, the variance-covariance matrix of $\hat{\boldsymbol{\beta}}$ is proportional to $\left(X^\top X/n\right)^{-1}$. Thus it is desirable to maximize the moment matrix $X^\top X/n$ in terms of, say, the conventional D-criterion:

$$\arg\max_{D_a} \left|X^\top X/n\right|^{1/p},$$

where $|\cdot|$ denotes the determinant of matrix. Note that $X^\top X = X_0^\top X_0 + X_a^\top X_a = pI_p + X_a^\top X_a$. By the Sylvester's determinant identity, $\left|X^\top X/p\right| = \left|I_{n-p} + X_a X_a^\top/p\right|$. As each element of X_a is within the interval $[-1, 1]$, it is easy to see that $X_a X_a^\top/p \le I_{n-p}$ in the Loewner order. It follows that

Theorem 7. *For the follow-up design problem, the combined design achieves the D-optimum if and only if $X_a X_a^\top = pI_{n-p}$. Such D-optimal combined design has a*

D-criterion of

$$\left|X^\top X/n\right|^{1/p} = \frac{p}{n} \cdot 2^{(n-p)/p}.$$

Note that $X_a X_a^\top = pI_{n-p}$ holds if and only if (i) each element of D_a equals 1 or -1 and (ii) the rows of X_a are orthogonal to each other. In particular, the equality holds when D_a is a fraction of Hadamard matrix.

Now consider the second criterion, minimizing the quantity $\sum_{i=1}^n G_{ii}^2$. We have derived a lower bound of this quantity among all possible follow-up designs:

Theorem 8. *For any follow-up design, it holds that*

$$\sum_{i=1}^{n} G_{ii}^2 \geq \frac{(n-p)n}{4p}, \tag{5.6}$$

where G_{ii}'s are the diagonal elements of the matrix $I_n - X(X^\top X)^{-1}X^\top$, X being the model matrix of the combined design.

When $(n, p) = (12, 8)$, such lower bound can be attained, for example, by

$$D_a = \begin{bmatrix} + & + & + & - & + & - & - & - \\ + & - & + & + & - & - & - & + \\ + & - & - & - & - & - & + & - \\ + & + & + & - & - & + & + & + \end{bmatrix},$$

with D_0 chosen as the first 8 columns of Design A in Table 5.1. It is notable that this D_a also satisfies the condition in Theorem 7, so that the combined design attains D-optimum and the lowest $\sum_{i=1}^n G_{ii}^2$ simultaneously.

For larger n and p, however, it is unknown to us whether the lower bound in Theorem 8 is attainable. In practice, we recommend searching for a design that is D-optimal and nearly attains the lower bound of $\sum_{i=1}^n G_{ii}^2$. A convenient construction

of such designs will be introduced later (see "D_{a3}" in Section 5.5.

Now consider the third criterion, minimizing $\left|\text{Cov}(\hat{\sigma}^2, \hat{\beta}_j)\right|$, $1 \le j \le k$. From Section 5.2, this criterion only matters when the error distribution is skewed, as $\left|\text{Cov}(\hat{\sigma}^2, \hat{\beta}_j)\right|$. Under skewed error distributions, Design A still achieves $\text{Cov}(\hat{\sigma}^2, \hat{\beta}_j) = 0$ for each $1 \le j \le k$, while no other follow-up designs achieve zero correlations, as suggested by numerical results. As such, Design A achieves the optimum under this correlation criterion among all follow-up designs. On the other hand, we have found that if a design is optimal in terms of the first two criterion, it is nearly-optimal in the correlation criterion:

Theorem 9. *In the follow-up design problem, suppose a combined design attains the D-optimum (in Theorem 2) as well as the lower bound of* $\sum_{i=1}^{n} G_{ii}^2$ *(in Theorem 3). Then under this design, for any error distribution, we have*

$$\sum_{j=1}^{k} corr^2(\hat{\beta}_j, \hat{\sigma}^2) < \frac{1}{4},$$

where $corr(,)$ *indicates the correlation.*

In other words, for any optimal design in terms of the first two criterion, the squared correlation between $\hat{\sigma}^2$ and a main effect is less that $1/(4k)$ on average. Note that $1/(4k)$ is the theoretical bound under any distribution and any design size. For common distributions, the (absolute) correlations are even much smaller, hence very close to 0.

As a summary, this section studies the theory of optimal follow-up design regarding three criteria: the D-criterion, minimizing $\text{Var}(\hat{\sigma}^2)$, and minimizing $\text{Cov}(\hat{\sigma}^2, \hat{\beta}_j)$. Explicit optimal values of the three criteria have been obtained. While the three criteria cannot be optimized simultaneously, we have shown that if a design is optimal in terms of the first two criteria, it is also favorable in terms of the third criterion.

Thus in practice, it suffices to optimize the design in terms of the first two criteria. In the next section we propose a convenient design construction for this purpose. We will also consider several follow-up designs suggested in the literature, and a comparison will be conducted across different designs.

5.5. The follow-up design problem: comparing several types of follow-up designs

In this section, we study four types of follow-up designs (D_a):

- D_{a0}: Center points, i.e., $D_{a0} = O$
- D_{a1}: Replicates one run in D_0 for $n-p$ times, i.e., $D_{a1} = \mathbf{1}_{n-p}\boldsymbol{d}^\top$, $\boldsymbol{d}$ being one run of D_0
- D_{a2}: Replicates $n-p$ runs in D_0, each for one time, i.e., $D_{a2} = [I_{n-p}, O]R_1D_0$, R_1 being a permutation matrix
- D_{a3}: Further permute the columns of D_{a2}, i.e., $D_{a3} = D_{a2}R_2$, R_2 being a permutation matrix

D_{a0} is the conventional follow-up design which yields the aforementioned Design A. D_{a1} and D_{a2} are two forms of *partial replication*. There exists a large body of literature on the topic of partial replication, see, e.g., Gilmour and Trinca (2012) and Leonard and Edwards (2017). Like the center points, partial replication is also widely used in practice, and here we evaluate the D- and $\sum_{i=1}^{n} G_{ii}^2$-criteria of two typical partial-replications, D_{a1} and D_{a2}. The last type of follow-up, D_{a3}, is proposed by us as a D-optimal and nearly minimal-$\sum_{i=1}^{n} G_{ii}^2$ follow-up design. We will show that D_{a3} is the most favorable among the four types of follow-ups.

First, in terms of the D-criterion:

Proposition 4. *The D-criterion of the combined design equals*

$$\begin{cases} (p/n)^{(p-1)/p} & \text{for } D_{a_0}; \\ (p/n)\cdot(n-p+1)^{1/p} & \text{for } D_{a_1}; \\ (p/n)\cdot 2^{(n-p)/p} & \text{for both } D_{a2} \text{ and } D_{a3}, \text{ with any choice of } R_1 \text{ and } R_2. \end{cases}$$

Note that $(p/n)^{(p-1)/p} < (p/n)\cdot(n-p+1)^{1/p} < (p/n)\cdot 2^{(n-p)/p}$, so D_{a0} is the least D-efficient while D_{a2} and D_{a3} are the most D-efficient. In fact, by Theorem 7, D_{a2} and D_{a3} achieve the D-optimality among follow-up designs. Yet (by permuting columns), D_{a3} achieves smaller $\sum_{i=1}^{n} G_{ii}^2$ than D_{a2}. Explicitly:

Theorem 10. *The quantity $\sum_{i=1}^{n} G_{ii}^2$ for the combined design equals*

$$\begin{cases} \dfrac{(n-p)(np-2p+1)}{np} & \text{for } D_{a0}; \\ \dfrac{(n-p)^2}{n-p+1} & \text{for } D_{a1}; \\ \dfrac{n-p}{2} & \text{for } D_{a2}, \text{ with any choice of } R_1; \end{cases}$$

And for D_{a3} with any choice of R_2, $\sum_{i=1}^{n} G_{ii}^2 \le (n-p)/2$.

Note that $(n-p)(np-2p+1)/(np) > (n-p)^2/(n-p+1) > (n-p)/2$, thus D_{a3} is the most favorable among the four types of designs under the criterion of $\sum_{i=1}^{n} G_{ii}^2$, with any choice of column permutation (R_2). Moreover, for many choices of R_2, D_{a3} actually attains nearly-optimal $\sum_{i=1}^{n} G_{ii}^2$, that is, it will approach the lower bound in Theorem 8. To see this, for $(n,p) = (12,8)$ we evaluate $100,000$ versions of D_{a3}, each with a randomly generated R_1 and R_2. (Note: D_0 is chosen as the first 8 columns of Design A in Table 5.1; the random permutations R_1 and R_2 are implemented by the

`sample` function in `R`). The $\sum_{i=1}^{n} G_{ii}^2$ of the $100,000$ D_{a3}'s are distributed as below:

$\sum_{i=1}^{n} G_{ii}^2$	1.625	1.6875	1.75	1.8125	2
Frequency	37388	32149	15724	10674	4065

Note that when $(n,p) = (12,8)$, the lower bound of $\sum_{i=1}^{n} G_{ii}^2$ in Theorem 8 is 1.5, while D_{a2} attains a $\sum_{i=1}^{n} G_{ii}^2$ of 2. As expected (from Theorem 10), the $\sum_{i=1}^{n} G_{ii}^2$ of D_{a3} is between those of D_{a2} and the lower bound. Moreover, it can be seen that most D_{a3}'s attain a $\sum_{i=1}^{n} G_{ii}^2$ that is quite close to the lower bound while considerably smaller than that of D_{a2}. This is a general phenomenon for larger n and p. Thus in practice, it is easy to find a D_{a3} that attains nearly optimal $\sum_{i=1}^{n} G_{ii}^2$ (by randomly generating and comparing a small number of D_{a3}'s).

5.6. Proofs

Lemma 10. *For the follow up design problem, G_{ii}'s ($1 \le i \le p$) are the diagonal elements of $p^{-2} X_0 X_a^\top \left(I_{n-p} + p^{-1} X_a X_a^\top\right)^{-1} X_a X_0^\top$, and G_{ii}'s ($p+1 \le i \le n$) are the diagonal elements of $I_{n-p} - p^{-1} X_a X_a^\top + p^{-2} X_a X_a^\top \left(I_{n-p} + p^{-1} X_a X_a^\top\right)^{-1} X_a X_a^\top$.*

Proof of Lemma 10. By the Woodbury matrix identity,

$$\begin{aligned}(X^\top X)^{-1} &= (X_0^\top X_0 + X_a^\top X_a)^{-1} = \left(p^{-1} I_p + X_a^\top X_a\right)^{-1} \\ &= (X_0^\top X_0)^{-1} - (X_0^\top X_0)^{-1} X_a^\top \left(I_{n-p} + X_a (X_0^\top X_0)^{-1} X_a^\top\right)^{-1} X_a (X_0^\top X_0)^{-1}\end{aligned}$$

In the following we assume that D_0 is a Hadamard matrix. That means $X_0^\top X_0 = p I_p$. Then

$$(X^\top X)^{-1} = p^{-1} I_p - p^{-2} X_a^\top \left(I_{n-p} + p^{-1} X_a X_a^\top\right)^{-1} X_a, \tag{5.7}$$

and therefore

$$X\left(X^\top X\right)^{-1}X^\top = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \tag{5.8}$$

where

$$A_{11} = I_p - p^{-2}X_0X_a^\top\left(I_{n-p} + p^{-1}X_aX_a^\top\right)^{-1}X_aX_0^\top, \tag{5.9}$$

$$A_{12} = p^{-1}X_0X_a^\top - p^{-2}X_0X_a^\top\left(I_{n-p} + p^{-1}X_aX_a^\top\right)^{-1}X_aX_a^\top,$$

$$A_{21} = p^{-1}X_aX_0^\top - p^{-2}X_aX_0^\top\left(I_{n-p} + p^{-1}X_aX_a^\top\right)^{-1}X_aX_0^\top,$$

$$A_{22} = p^{-1}X_aX_a^\top - p^{-2}X_aX_a^\top\left(I_{n-p} + p^{-1}X_aX_a^\top\right)^{-1}X_aX_a^\top. \tag{5.10}$$

The result follows from the expressions of A_{11} and A_{22}. □

Proof of Theorem 9. From the proof of Theorem 8, a combined design attains the D-optimum as well as the lower bound of $\sum_{i=1}^n G_{ii}^2$ if and only if:

$$X_aX_a^\top = pI_{n-p}, \tag{5.11}$$

$$G_{ii} = \frac{n-p}{2p}, \text{ for any } 1 \le i \le p, \text{ and} \tag{5.12}$$

$$G_{ii} = \frac{1}{2}, \text{ for any } p+1 \le i \le n. \tag{5.13}$$

We first derive $(X^\top X)^{-1}X^\top[G_{11},\ldots,G_{nn}]^\top$ for such designs, in view of Lemma 2. In view of equations (5.7) and (5.11),

$$\left(X^\top X\right)^{-1} = p^{-1}I_{n-p} - p^{-2}X_a^\top\left(I_{n-p} + p^{-1}pI_{n-p}\right)^{-1}X_a = p^{-1}I_{n-p} - (2p^2)^{-1}X_a^\top X_a, \tag{5.14}$$

and therefore

$$\begin{aligned}&(X^\top X)^{-1}X^\top = (X^\top X)^{-1}[X_0^\top, X_a^\top]\\ =&\Big[p^{-1}X_0^\top - (2p^2)^{-1}X_a^\top X_a X_0^\top, p^{-1}X_a^\top - (2p^2)^{-1}X_a^\top X_a X_a^\top\Big].\end{aligned}$$

On the other hand, conditions (5.12) and (5.13) indicate that $[G_{11},\ldots,G_{nn}] = [(n-p)/(2p)\cdot\mathbf{1}_p^\top, (1/2)\mathbf{1}_{n-p}^\top]$. Thus

$$\begin{aligned}&(X^\top X)^{-1}X^\top[G_{11},\ldots,G_{nn}]^\top\\ =&(n-p)/(2p)\Big[p^{-1}X_0^\top - (2p^2)^{-1}X_a^\top X_a X_0^\top\Big]\mathbf{1}_p + (1/2)\Big[p^{-1}X_a^\top - (2p^2)^{-1}X_a^\top X_a X_a^\top\Big]\mathbf{1}_{n-p}\\ =&(n-p)/(2p^2)X_0^\top\mathbf{1}_p - (n-p)/(4p^3)X_a^\top X_a X_0^\top\mathbf{1}_p + (2p)^{-1}X_a^\top\mathbf{1}_{n-p} - (4p)^{-1}X_a^\top\mathbf{1}_{n-p}\\ =&(n-p)/(2p^2)X_0^\top\mathbf{1}_p - (n-p)/(4p^3)X_a^\top X_a X_0^\top\mathbf{1}_p + (4p)^{-1}X_a^\top\mathbf{1}_{n-p}.\end{aligned}$$

As X_0 is a Hadamard matrix with the first column being ones, $X_0^\top\mathbf{1}_p = [p,0,\ldots,0]^\top$. Then since the first column of X_a is $\mathbf{1}_{n-p}$, we have $X_a X_0^\top\mathbf{1}_p = X_a[p,0,\ldots,0]^\top = p\mathbf{1}_{n-p}$. It follows that

$$\begin{aligned}&(X^\top X)^{-1}X^\top[G_{11},\ldots,G_{nn}]^\top\\ =&(n-p)/(2p^2)[p,0,\ldots,0]^\top + \left(\frac{1}{4p} - \frac{n-p}{4p^2}\right)X_a^\top\mathbf{1}_{n-p}.\end{aligned}$$

Denote $X_a^\top\mathbf{1}_{n-p} = \boldsymbol{\gamma} = (\gamma_1,\gamma_2,\ldots,\gamma_p)^\top$, then for $1\le j\le k$, in view of Lemma 2,

$$\mathrm{Cov}(\hat\beta_j,\hat\sigma^2) = \frac{E(\epsilon_1^3)}{n-p}\cdot\frac{2p-n}{4p^2}\gamma_{j+1}.$$

Apparently, $\gamma_1 = n - p$, and note that

$$\begin{aligned}\sum_{s=1}^{p} \gamma_s^2 &= \left(X_a^\top \mathbf{1}_{n-p}\right)^\top X_a^\top \mathbf{1}_{n-p} \\ &= \mathbf{1}_{n-p}^\top X_a X_a^\top \mathbf{1}_{n-p} \\ &= \mathbf{1}_{n-p}^\top (pI_{n-p}) \mathbf{1}_{n-p} \\ &= p(n-p),\end{aligned}$$

then $\sum_{s=2}^{p} \gamma_s^2 = p(n-p) - (n-p)^2 = (2p-n)(n-p)$, and then

$$\begin{aligned}\sum_{j=1}^{k} \mathrm{Cov}^2(\hat{\beta}_j, \hat{\sigma}^2) &= \left[\frac{E(\epsilon_1^3)}{n-p} \cdot \frac{2p-n}{4p^2}\right]^2 \sum_{j=1}^{k} \gamma_{j+1}^2 \\ &= \frac{\left[E(\epsilon_1^3)\right]^2 (2p-n)^3}{4p^2(n-p)}.\end{aligned}$$

By equation (5.14), in view that each element of X_a is 1 or -1, it is evident that each diagonal element of $(X^\top X)^{-1}$ equals $p^{-1} - (2p^2)^{-1}(n-p) = (3p-n)/(2p^2)$, and therefore $\mathrm{Var}(\hat{\beta}_j) = \sigma^2(3p-n)/(2p^2)$. As $\sum_{i=1}^{n} G_{ii}^2$ attains the lower bound of $(n-p)n/(4p)$, by Lemma 1, we have $\mathrm{Var}(\hat{\sigma}^2) = \sigma^4\left\{2/(n-p) + \mathrm{EK} \cdot n/[4p(n-p)]\right\}$. Hence

$$\begin{aligned}\sum_{j=1}^{k} \mathrm{corr}^2(\hat{\beta}_j, \hat{\sigma}^2) &= \sum_{j=1}^{k} \frac{\mathrm{Cov}^2(\hat{\beta}_j, \hat{\sigma}^2)}{\mathrm{Var}(\hat{\sigma}^2)\,\mathrm{Var}(\hat{\beta}_j)} \\ &= \frac{\left[E(\epsilon_1^3)\right]^2 (2p-n)^3}{4p^2(n-p)} \Bigg/ \left[\sigma^6 \cdot \frac{3p-n}{2p^2} \cdot \frac{8p + n \cdot \mathrm{EK}}{4p(n-p)}\right] \\ &= \frac{\left[E(\epsilon_1^3)\right]^2}{\sigma^6} \cdot \frac{(2p-n)^3}{2p(3p-n)(n\delta + 8p)},\end{aligned}$$

where δ denotes the EK. By Cauchy-Schwartz inequality, $\left[E(\epsilon_1^3)\right]^2 \le E(\epsilon_1^4)E(\epsilon_1^2) = \sigma^6(\delta+3)$. As $p > n/2$, we have $(8p)/n > 3$, and thus $(\delta+3)/(n\delta+8p)$ is an increasing

function as δ increases and $(\delta+3)/(n\delta+8p) < 1/n$ Hence

$$\begin{aligned}\sum_{j=1}^{k}\operatorname{corr}^2(\hat{\beta}_j,\hat{\sigma}^2) &\le \frac{\sigma^6(\delta+3)}{\sigma^6}\cdot\frac{(2p-n)^3}{2p(3p-n)(n\delta+8p)}\\ &= \frac{(2p-n)^3}{2p(3p-n)}\cdot\frac{\delta+3}{n\delta+8p}\\ &< \frac{(2p-n)^3}{2np(3p-n)} = \frac{(2r-1)^3}{2r(3r-1)},\end{aligned}$$

where $r = p/n$. It is easy to see that $(2r-1)^3\Big/\big[2r(3r-1)\big]$ is increasing as r increases in $(1/2,1)$. Thus $\sum_{j=1}^{k}\operatorname{corr}^2(\hat{\beta}_j,\hat{\sigma}^2) < (2-1)^3/[2(3-1)] = 1/4$. □

Proof of Proposition 4. For D_{a0}, the combined design is Design A (in Section 5.2), and thus $X^\top X$ is a diagonal matrix with diagonal elements being $(n,p,...,p)$. Thus its D-criterion follows.

For D_{a1}, the combined model matrix equals $X = \big[X_0^\top,\boldsymbol{x},...,\boldsymbol{x}\big]$, where $\boldsymbol{x}$ is one run for X_0. Thus $X^\top X = [X_0^\top,\boldsymbol{x},...,\boldsymbol{x}][X_0^\top,\boldsymbol{x},...,\boldsymbol{x}]^\top = pI_p + (n-p)\boldsymbol{x}\boldsymbol{x}^\top$. By Sylvester's equality, $\big|X^\top X/p\big| = \big|I_p + \boldsymbol{x}\boldsymbol{x}^\top\cdot(n-p)/p\big| = \big|1+\boldsymbol{x}^\top\boldsymbol{x}\cdot(n-p)/p\big| = 1+n-p$. Thus the D-criterion equals $\big|X^\top X/n\big|^{1/p} = \big[(1+n-p)\big/(n/p)^p\big]^{1/p} = (p/n)\cdot(n-p+1)^{1/p}$.

By Theorem 7, both D_{a2} and D_{a3} give D-optimal combined designs, thus the D-criterion follows. □

Proof of Theorem 10. For D_{a0}, the combined design is Design A (in Section 5.2), and its $\sum_{i=1}^{n} G_{ii}^2$ is given in Theorem 1.

Recall that G_{ii}'s are the diagonal elements of $I_n - X(X^\top X)^{-1}X^\top$. By equation (5.8), it suffices to evaluate A_{11} and A_{22}, which are expressed in equations (5.9) and (5.10).

For D_{a1}, $X_{a1} = \mathbf{1}_{n-p}\boldsymbol{x}^\top$ with $\boldsymbol{x}^\top$ being one run of X_0. Thus $X_{a1}X_{a1}^\top = p\mathbf{1}_{n-p}\mathbf{1}_{n-p}^\top$,

and then $\left(I_{n-p}+p^{-1}X_{a1}X_{a1}^\top\right)^{-1} = \left(I_{n-p}+\mathbf{1}_{n-p}\mathbf{1}_{n-p}^\top\right)^{-1} = I_{n-p}-\mathbf{1}_{n-p}\mathbf{1}_{n-p}^\top/(n-p+1)$. Thus

$$\begin{aligned} X_{a1}^\top\left(I_{n-p}+p^{-1}X_{a1}X_{a1}^\top\right)^{-1}X_{a1} &= X_{a1}^\top X_{a1} - \frac{1}{n-p+1}X_{a1}^\top\mathbf{1}_{n-p}\mathbf{1}_{n-p}^\top X_{a1} \\ &= \boldsymbol{x}\mathbf{1}_{n-p}^\top\mathbf{1}_{n-p}\boldsymbol{x}^\top - \frac{1}{n-p+1}\boldsymbol{x}\mathbf{1}_{n-p}^\top\mathbf{1}_{n-p}\mathbf{1}_{n-p}^\top\mathbf{1}_{n-p}\boldsymbol{x}^\top \\ &= (n-p)\boldsymbol{x}\boldsymbol{x}^\top - \frac{(n-p)^2}{n-p+1}\boldsymbol{x}\boldsymbol{x}^\top \\ &= \frac{n-p}{n-p+1}\boldsymbol{x}\boldsymbol{x}^\top. \end{aligned} \tag{5.15}$$

Thus $X_0X_{a1}^\top\left(I_{n-p}+p^{-1}X_{a1}X_{a1}^\top\right)^{-1}X_{a1}X_0^\top = X_0\boldsymbol{x}\boldsymbol{x}^\top X_0\cdot(n-p)/(n-p+1)$. Suppose $\boldsymbol{x}$ is the i^*th run of X_0, then $X_0\boldsymbol{x}\boldsymbol{x}^\top X_0 = (p\boldsymbol{e}_{i^*})(p\boldsymbol{e}_{i^*})^\top = p^2\boldsymbol{e}_{i^*}\boldsymbol{e}_{i^*}^\top$, where $\boldsymbol{e}_{i^*}$ is a vector with the i^*th element being 1 and all other elements being 0. Thus for D_{a1},

$$\begin{aligned} A_{11} &= I_p - p^{-2}X_0X_{a1}^\top\left(I_{n-p}+p^{-1}X_{a1}X_{a1}^\top\right)^{-1}X_{a1}X_0^\top \\ &= I_p - \frac{n-p}{n-p+1}\boldsymbol{e}_{i^*}\boldsymbol{e}_{i^*}^\top. \end{aligned}$$

Note that G_{ii}'s $(1\le i\le p)$ are the diagonal elements of $I_p - A_{11}$, thus $G_{i^*i^*} = (n-p)/(n-p+1)$, and for any $1\le i\le p$, $i\ne i^*$, $G_{ii}=0$. On the other hand, by equation (5.15), $X_aX_{a1}^\top\left(I_{n-p}+p^{-1}X_{a1}X_{a1}^\top\right)^{-1}X_{a1}X_a^\top = (\mathbf{1}_{n-p}\boldsymbol{x}^\top)\boldsymbol{x}\boldsymbol{x}^\top(\boldsymbol{x}\mathbf{1}_{n-p}^\top)\cdot(n-p)/(n-p+1) = \mathbf{1}_{n-p}(\boldsymbol{x}^\top\boldsymbol{x})(\boldsymbol{x}^\top\boldsymbol{x})\mathbf{1}_{n-p}^\top\cdot(n-p)/(n-p+1) = \mathbf{1}_{n-p}\mathbf{1}_{n-p}^\top\cdot p^2(n-p)/(n-p+1)$. Thus for D_{a1},

$$\begin{aligned} A_{22} &= p^{-1}X_{a1}X_{a1}^\top - p^{-2}X_{a1}X_{a1}^\top\left(I_{n-p}+p^{-1}X_{a1}X_{a1}^\top\right)^{-1}X_{a1}X_{a1}^\top \\ &= \mathbf{1}_{n-p}\mathbf{1}_{n-p}^\top - \frac{n-p}{n-p+1}\mathbf{1}_{n-p}\mathbf{1}_{n-p}^\top. \end{aligned}$$

Note that G_{ii}'s $(p+1\le i\le n)$ are the diagonal elements of $I_{n-p} - A_{22}$, we have $G_{ii} = (n-p)/(n-p+1)$, for any $p+1\le i\le n$. Thus for the combined design from D_{a1}, $\sum_{i=1}^n G_{ii}^2 = \left[(n-p)/(n-p+1)\right]^2 + (n-p)\cdot\left[(n-p)/(n-p+1)\right]^2 = (n-p^2)/(n-p+1)$. □

Chapter 6

Finding optimal run orders in design of experiments

6.1. Introduction

Many scientific and industrial experiments are carried out in a sequence. For an n–run design, there are $n!$ possible run orders to conduct the experiment. The conventional manner is to randomize the run order. Cox (1951) first pointed out the need of a systematic run order rather than a random one, when the observations are affected by a time trend. Cox showed that in the presence of a time trend, different run orders will result in considerably different efficiency in terms of estimating factor effects. A systematic run order, which minimizes the bias due to time trend, is then called for. Besides, a systematic run order is also desired when it is costly or time-consuming to switch the level of experimental factors (Draper and Stoneman 1968).

Remark 5. The run order problem here is different from the order-of-addition problem discussed in Chapters 2-4. In order-of-addition experiments, each experimental

unit is an order of some components, and our goal was to choose some orders as experimental runs. On the contrary, the run order problem here occurs when the experimental runs are already chosen, and our goal is to carry out these runs in an appropriate sequence.

There has been much literature on the choice of systematic run orders. Draper and Stoneman (1968) considered all 8-run two-level factorial designs. They found the run orders for which the factor effects are least correlated with a linear time trend, as well as the run orders that requires the least number of level-changes between consecutive runs. Joiner and Campbell (1976) presented real examples to emphasize the importance of run order. Coster and Cheng (1988) provided the solutions of minimum level-change run orders among trend-free sequences for fractional factorials. Cheng and Steinberg (1991) introduced trend-robust run orders of two-level factorials.

We are mainly interested in finding an optimal run order which minimizes the costs of level-changes. Cheng et al. (1998) thoroughly discussed how to construct run orders with minimum (or maximum) number of level changes, for two-level and some three-level factorial designs. Quinlan and Lin (2015) studied the run order problem for Plackett-Burman designs. However, these approaches are all limited to designs with specific structures. We propose an algorithm to construct optimal run orders for any design of experiment with respect to flexible distance criterion (based on the number of level changes). Our algorithm is based on the observation that the run order optimization is essentially a Travelling Salesman Problem (TSP). We illustrate how to convert the run order problem into a TSP with a modification, and how to solve the optimal run order using TSP solvers. Compared with existing approaches (such as Cheng et al. 1998), our method is much more widely applicable.

We have found a variety of new optimal run orders, and some of them are tabulated for practical use.

The remainder of this chapter is organized as follows. Section 6.2 illustrates the need to optimize the run order via a 2^3 design example. Our optimization algorithm is then proposed in Section 6.3. Section 6.4 applies the algorithm to obtain minimum-level-change run orders for two-level designs, and displays the results for practical use. Some theoretical results are provided in Section 6.5. Section 6.6 discusses how to apply the proposed algorithm to general (such as, higher- or mixed-level) designs as well as general criteria.

6.2. An illustrative example

Consider the run order problem in a 2^3 full factorial experiment. The design consists of three factors and eight different experimental runs. Write the three factors as A, B, and C; each factor has two levels, denoted by "−" and "+". The eight experimental runs can be written as "(1)", "a", "b", "c", "ab", "ac", "bc", and "abc", where the presence of a letter indicates that the specified factor is at its "+" level and the absence of a letter indicates that the specified factor is at its "−" level. For example, "ab" indicates that factor A, B are at "+" and C is at "−", namely, $(A, B, C) = (+, +, -)$; and the run "(1)" indicates that all factors are set at "−".

A run order is defined as any permutation of these eight runs, and is written as an ordered sequence. For example, the conventional Yate's order is represented as $\{(1), a, b, ab, c, ac, bc, abc\}$, which indicates that run "(1)" is conducted first, followed by the run "a", ..., and finally the run "abc". In fact, there are a total of $8! = 40320$ possible run orders. Table 6.1 shows some potential run orders: Table 6.1(a) is the Yates order; Table 6.1(b) is an example from Draper and Stoneman (1968); Table

6.1(c) is generated via randomization; and Table 6.1(d) is the optimal run order as we will explain later.

To evaluate the run order issue, there are different criteria based on practical needs. A wide class of criteria is based upon the level changes between consecutive runs: when it is costly or time-consuming to reset experimental factors, one desires to reduce the number of level changes. Another class of criteria is related to the time-trend: in some experiments, the response is affected by a time trend effect. The time-trend effect may be a linear, higher-order polynomial fixed effect; an auto-regressive model or other time series models. In this case, we desire to choose a run order so that factor effects are least confounded with the time trend. The choice of optimality criteria has been discussed in previous work, such as Draper and Stoneman (1968), Joiner and Campbell (1976), Wang and Jan (1995), just to name a few.

For illustration purpose, here we use the "total number of level changes" as the optimality criterion. It is denoted as LC throughout this chapter. Between any two consecutive runs, the level-change is the number of factors at different levels across these two runs. For example, the level-change number between (1) and a is one, between a and b is 2, and between ab and c is 3. For any specific run order, the (total) distance LC is defined as the sum of level-change numbers between all pairs of consecutive runs. As an example, LC of the order $\{1, a, b, ab, c, ac, bc, abc\}$ is $1+2+1+3+1+2+1 = 11$. A formal definition of LC will be given in Section 4.5. The smaller LC is, the better (see, for example, Draper and Stoneman 1968 and Cheng et al. 1998).

For the run orders of designs given in Table 6.1: LC = 11 for the Yates order; LC = 9 for the one in Draper and Stoneman (1968); and LC = 13 for the randomized run order. Table 6.1(d) shows the optimal run order, and this arrangement requires only 7 level changes.

The LC numbers for all 8! possible sequences are evaluated and displayed in Figure 1, it varies from 7 to 17, with the average being 12 and standard deviation being 1.85. Thus the optimal run order reduces the LC by almost half of the average LC. This is quite favorable in practice when the level-changes are costly or time-consuming. Figure 1 also indicates that there is only a small proportion (0.36%) among all possible orders that achieve the minimum LC. This implies that the optimal order is unlikely to be obtained via a random search. Here, we seek an efficient way to obtain the optimal run order.

Table 6.1: Four run orders of the 2^3 full factorial design

(a) Yate's Order

	A	B	C
(1)	−	−	−
a	+	−	−
b	−	+	−
ab	+	+	−
c	−	−	+
ac	+	−	+
bc	−	+	+
abc	+	+	+

(b) Run order in Draper and Stoneman (1968)

	A	B	C
(1)	−	−	−
ab	+	+	−
abc	+	+	+
ac	+	−	+
c	−	−	+
bc	−	+	+
b	−	+	−
a	+	−	−

(c) Randomized run order

	A	B	C
b	−	+	−
c	−	−	+
ac	+	−	+
(1)	−	−	−
ab	+	+	−
abc	+	+	+
a	+	−	−
bc	−	+	+

(d) Optimal run order

	A	B	C
ab	+	+	−
b	−	+	−
bc	−	+	+
c	−	−	+
(1)	−	−	−
a	+	−	−
ac	+	−	+
abc	+	+	+

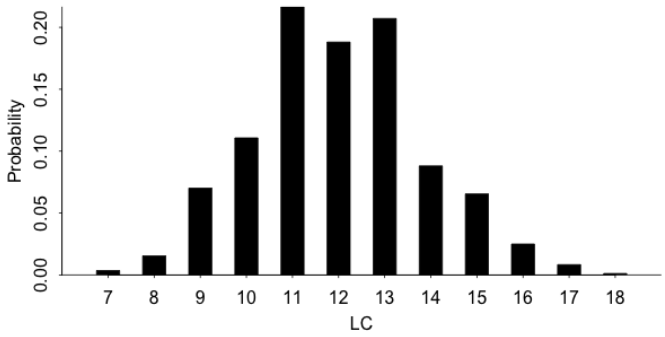


Figure 6.1: Distribution of LC among all possible run orders for the 2^3 design

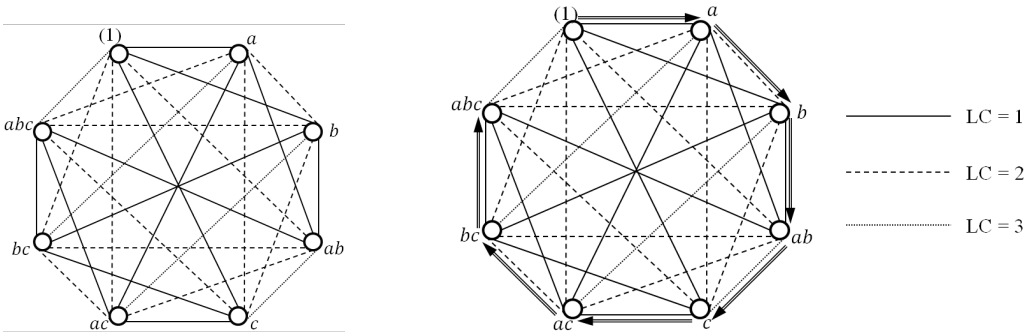
6.3. Proposed algorithm

Given a design and any arbitrary distance measurement between each pair of runs, an algorithm is proposed here to generate an optimal run order which minimizes the total distance between consecutive runs. We first illustrate the complete procedure using the example in Section 4.2, and then summarize the proposed algorithm.

To obtain the optimal run order, we model this problem in a graph. Treat each run as a node, define the weight of edge between any two nodes as the LC distance between these two nodes. Consider once again the 2^3 design in Section 2, its "distance graph" is then constructed, as shown in Figure 5.2 (a).

Any run order is now visualized as a path in this graph, and LC for a specific order is simply the total length of the corresponding path. For example, the Yate's order $\{(1), a, b, ab, c, ac, bc, abc\}$ corresponds to the route in Figure 5.2 (b), and its LC number is 11.

Specifically, a possible run order is an unclosed path which visits each node exactly once. Among such paths, we desire to find a shortest one, corresponding to an optimal run order. This is similar to the famous Traveling Salesman Problem (TSP). See the



(a) Distance graph

(b) Representation of the Yate's order on the distance graph

Figure 6.2: The distance graph and the representation of a run order on it

appendix for a brief introduction of TSP. However, there is a slight difference: a TSP solution gives a shortest loop which visits all nodes exactly once and returns to the initial starting node, while the run order problem here is seeking a path that does not need to return to the initial node.

To convert the design run order problem into TSP, we add one virtual node V to the distance graph, where V has zero distance to all other nodes. The modified graph is shown in Figure 3. Then any loop L in the modified graph corresponds to a run order that starts from the next node of V and ends at the previous node of V in L. For example, the loop $V \to (1) \to a \to b \to ab \to c \to ac \to bc \to abc \to V$ corresponds to the Yate's order $\{(1), a, b, ab, c, ac, bc, abc\}$. Since V is zero-distant from other nodes, any loop has the same length as its corresponding run order.

An optimal run order is now equivalent to a shortest loop in the modified distance graph. A shortest loop in Figure 3 obtained by TSP solvers is $V \to ab \to b \to bc \to c \to (1) \to a \to ac \to abc \to V$, and therefore a minimum-LC run order is $\{ab, b, bc, c, (1), a, ac, abc\}$, as displayed in Table 6.1 (d).

We now present the complete algorithm. This algorithm is capable of producing an optimal run order, given *any* design and *any* distance measure.

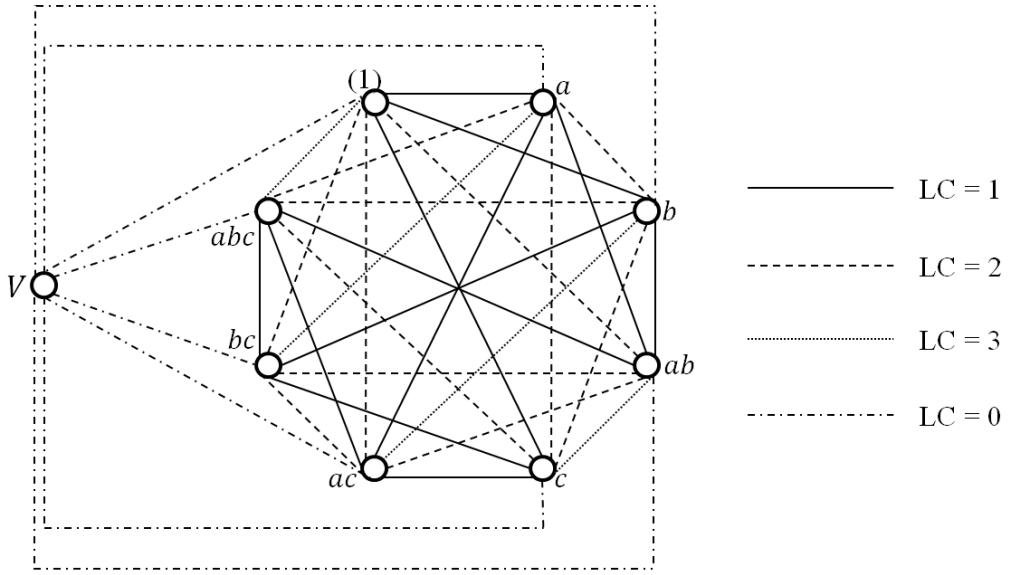


Figure 6.3: The modified distance graph

Step 1 Build up the distance graph: (a) Treat each run as a node, and (b) Define the length of edge as the distance between each pair of runs (e.g., the level-change number).

Step 2 Modify the distance graph by adding a virtual node which has zero distance from any other node.

Step 3 Apply Traveling Salesman Problem (TSP) solvers to find a shortest loop in the modified distance graph, and then retrieve the optimal run order.

Illustrative Example

Consider the 2^4 full factorial design (Table 6.2), with the four factors denoted by A, B, C, and D. The goal is to find a minimum-LC run order, and the algorithm is carried out in the following procedure.

Step 1. Set up a distance graph with the 16 runs as the nodes. In practice, the adjacency matrix of distance graph is necessary. For a graph with n nodes, its

Table 6.2: The 2^4 full factorial design

Run	A	B	C	D
(1)	−	−	−	−
a	+	−	−	−
b	−	+	−	−
ab	+	+	−	−
c	−	−	+	−
ac	+	−	+	−
bc	−	+	+	−
abc	+	+	+	−
d	−	−	−	+
ad	+	−	−	+
bd	−	+	−	+
abd	+	+	−	+
cd	−	−	+	+
acd	+	−	+	+
bcd	−	+	+	+
$abcd$	+	+	+	+

adjacency matrix is defined as an $n \times n$ matrix, in which the (i, j) entry is the distance from node i to j. The adjacency matrix of distance for 2^4 design is shown in Table 6.3 (tentatively ignore the last row and the last column of V).

Table 6.3: Adjacency matrix of the LC distance graph for the 2^4 design

	(1)	***a***	***b***	***ab***	***c***	***ac***	***bc***	***abc***	***d***	***ad***	***bd***	***abd***	***cd***	***acd***	***bcd***	***abcd***	***V***
(1)	0	1	1	2	1	2	2	3	1	2	2	3	2	3	3	4	**0**
a	1	0	2	1	2	1	3	2	2	1	3	2	3	2	4	3	**0**
b	1	2	0	1	2	3	1	2	2	3	1	2	3	4	2	3	**0**
ab	2	1	1	0	3	2	2	1	3	2	2	1	4	3	3	2	**0**
c	1	2	2	3	0	1	1	2	2	3	3	4	1	2	2	3	**0**
ac	2	1	3	2	1	0	2	1	3	2	4	3	2	1	3	2	**0**
bc	2	3	1	2	1	2	0	1	3	4	2	3	2	3	1	2	**0**
abc	3	2	2	1	2	1	1	0	4	3	3	2	3	2	2	1	**0**
d	1	2	2	3	2	3	3	4	0	1	1	2	1	2	2	3	**0**
ad	2	1	3	2	3	2	4	3	1	0	2	1	2	1	3	2	**0**
bd	2	3	1	2	3	4	2	3	1	2	0	1	2	3	1	2	**0**
abd	3	2	2	1	4	3	3	2	2	1	1	0	3	2	2	1	**0**
cd	2	3	3	4	1	2	2	3	1	2	2	3	0	1	1	2	**0**
acd	3	2	4	3	2	1	3	2	2	1	3	2	1	0	2	1	**0**
bcd	3	4	2	3	2	3	1	2	2	3	1	2	1	2	0	1	**0**
abcd	4	3	3	2	3	2	2	1	3	2	2	1	2	1	1	0	**0**
V	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Step 2. Add one row of zeros, as well as one column of zeros onto the above matrix, we obtain the adjacency matrix of modified distance graph. The modified

graph includes one more virtual node V than the distance graph, where V is zero-distant from all other nodes. This is a critical step of applying TSP.

Step 3. Input the adjacency matrix of modified graph into a TSP solver. *Concorde* (Applegate et al. 2003) is employed here as the TSP solver, as will be discussed in the Section 5.6.2. In this example, the solver provides the loop $V \to cd \to acd \to ac \to abc \to ab \to b \to bc \to c \to (1) \to a \to ad \to d \to bd \to bcd \to abcd \to abd \to V$. Therefore, an optimal run order for the 2^4 design is: $\{cd, acd, ac, abc, ab, b, bc, c, (1), a, ad, d, bd, bcd, abcd, abd\}$. This sequence has 15 level changes.

We end this section with a remark on the TSP solver.

In Step-3 of the proposed algorithm, we employ "Concorde" as our TSP solver. The Traveling Salesman Problem is $NP-$hard theoretically (Karp 1972), which means any TSP solver would require a long running time under the "worst case". Concorde is efficient in the sense that the algorithm terminates in a polynomial time under most situations (Applegate et al. 2011). Recent simulation results have supported this. For example, Hoos and Stützle (2014) showed that for a 500−node TSP, the average running time of Concorde is below 100 CPU seconds, and for 100 nodes, the average time is below one second. Note that industrial experiments typically involve at most a few hundred runs. Therefore, using the proposed algorithm, the optimal run order problem can be converted to a TSP with at most a few hundred nodes. That is computationally affordable, based on the the simulation results in Hoos and Stützle (2014). In fact, we have implemented the proposed algorithm on on a computer with a 2.7 GHz, Intel Core i5 processor, and a memory of 8 GB. To solve the minimum-LC run order of a 2^7 design (128 runs), the program took 1.9 seconds to run. For the 2^8 design (256 runs), the program took 11.4 seconds to run. In all the examples of this chapter, the algorithm took no more than 1 minute to

solve the optimal run order. In summary, our algorithm (using Concorde) is ensured to obtain an optimal result in a rather efficient manner.

6.4. Designs with minimum-level-change run orders

Applying the proposed algorithm in last section with the LC criterion (see Section 6.2), this section obtains "minimum-level-change" run orders for different types of designs. This class of run orders have been widely discussed in the literature. See, for example, Draper and Stoneman (1968), Cheng (1990), Cheng et al. (1998), and Quinlan and Lin (2015).

The proposed algorithm does not depend on any specific design structure, and is able to solve minimum-LC run orders for all different designs. For illustration, this section focuses on two-level designs: full factorial, regular fractional factorial, Plakett-Burman, and $D-$optimal designs. The first three are orthogonal and the fourth one is non-orthogonal. The run order problem for the first three classes were studied before; for these designs, our optimization results are consistent with the existing ones. (For each design, our optimal run order achieves the same LC as that of the existing result.) In the last subsection we solve and display optimal run orders for a class of $D-$optimal designs, and the results are apparently new in the literature. (Note that D-optimal designs are not unique. The results provided in Section 4.4 serve as a special class of $D-$optimal design, although the algorithm works for any $D-$optimal design.) More general cases are discussed in Section 6.

Note that all factors are assumed to be equally difficult (cost) to change levels in our work. If there is one (or more) factor that is hard-to-change (e.g. split-plot

designs), a large literature is devoted in this subject (see, for example, L. and Lucas 2002 and Jones and Nachtsheim 2009).

6.4.1. Two-level full factorial designs

The minimum-LC run orders for 2^3 and 2^4 full factorial are known (Cheng 1990 and Lin and W. 1997). Recall that an optimal run order for 2^3 design is $\{ab, b, bc,\ c, (1),\ a, ac, abc\}$, which has 7 level changes; for 2^4 design an optimal run order is $\{cd, acd, ac,\ abc, ab,\ b, bc, c, (1),\ a, ad, d, bd,\ bcd, abcd, abd\}$, which has 15 level changes. For the 2^5 factorial, an optimal run order is $\{e, ce, c,\ cd, acd, ad,\ a, ac,\ ace, ae, ade,\ acde, cde, de,\ d, (1), b, ab, abe, abce,\ abc, abcd, abd, bd, bcd,\ bc, bce, be, bde,\ abde, abcde, bcde\}$, which requires 31 level changes. These can be obtained via the proposed algorithm.

In general for a 2^k design, the proposed algorithm gives an optimal solution with LC being $2^k - 1$. This is consistent with Cheng et al. (1998), and Lin and W. (1997). As will be seen in Theorem 2 (Section 5), the average LC is $k2^{k-1}$ for the 2^k design. Recall that the minimum LC is $2^k - 1$, this is only a small fraction (about $2/k$) of the average LC. This indicates the optimal run order is quite preferable compared with a randomized one.

6.4.2. Two-level regular fractional factorial designs

Consider run orders in a general 2^{k-p} fractional factorials. Take a 2^{6-2} resolution IV design as an example: the 16−run with 6−factor design is given in Table 6.4. By using the proposed algorithm, we can obtain the optimal run order $\{acd, ac, bc,\ bcd, bdf, bf,\ af, adf,\ cdef, cef,\ abcef,\ abcdef,\ abde, abe,\ e, de\}$. It requires 23 level-changes, which matches the existing result for two-level fractional factorials in Cheng et al. (1998). The average LC is 48 among all run orders for this design (see Theorem 2 in Section

5), so the optimal run order again reduces the LC by half of the average.

Table 6.4: A 2^{6-2} resolution IV design

Run	A	B	C	D	E	F
1	−	−	−	−	+	−
2	+	−	−	−	−	+
3	−	+	−	−	−	+
4	+	+	−	−	+	−
5	−	−	+	−	+	+
6	+	−	+	−	−	−
7	−	+	+	−	−	−
8	+	+	+	−	+	+
9	−	−	−	+	+	−
10	+	−	−	+	−	+
11	−	+	−	+	−	+
12	+	+	−	+	+	−
13	−	−	+	+	+	+
14	+	−	+	+	−	−
15	−	+	+	+	−	−
16	+	+	+	+	+	+

Wu and Hamada (2011) summarized a catalog of small run 2–level fractional factorials. Here we have solved the optimal run orders for all 8–run and 16–run designs in their catalog, and the results are displayed in Table 6.5. The first two columns in Table 6.5 indicate each design with the name and generators, following the notations used in Wu and Hamada (2011). The third column shows the minimum LC obtained by the proposed algorithm, and as a comparison, the corresponding average LC is given in the fourth column. For practical use, a sample optimal run order is provided in the last column. Note that the minimum LC is the same as the average LC for some designs. There is a theoretical reason behind this (see, Cheng et al. (1998)).

Table 6.5: Minimum-LC run orders for 8–run and 16–run fractional factorials

Design	Generators	Minimum LC	Average LC	Sample optimal run order
2^{4-1}_{IV}	$4 = 123$	14	16	*cd, bd, abcd, ab, ad, ac, bc, (1)*
2^{5-2}_{III}	$4 = 123$, $5 = 13$	15	20	*abcde, abd, a, ace, de, cd, bc, be*
2^{6-3}_{III}	$4 = 123$, $5 = 13$, $6 = 23$	21	24	*abd, be, ace, cd, bcf, abcdef, def, af*
2^{7-4}_{III}	$4 = 123$, $5 = 13$, $6 = 23$, $7 = 123$	28	28	Any order*
2^{5-1}_{V}	$5 = 1234$	30	40	*cde, ace, abc, acd, bcd, abd, d, b, bce, c, e, a, ade, abcde, bde, abe*
2^{6-2}_{IV}	$5 = 123$, $6 = 124$	31	48	*bef, aef, acf, bcf, bcd, acd, ade, bde, ce, (1), ab, abdf, df, cdef, abcdef, abce*
2^{6-2}_{III}	$5 = 12$, $6 = 134$	31	48	*abce, abef, abde, abcdef, acdf, ad, af, ac, bcd, bdf, bcf, b, e, cde, def, cef*
2^{7-3}_{IV}	$5 = 123$, $6 = 124$, $7 = 134$	45	56	*aefg, acf, bcfg, abcdefg, abdf, dfg, acdg, ceg, abce, bcd, cdef, bef, bdeg, ade, (1), abg*
2^{8-4}_{IV}	$5 = 123$, $6 = 124$, $7 = 134$, $8 = 234$	60	64	*abgh, aefg, abcdefgh, bcdh, bcfg, abce, abdf, dfgh, adeh, cdef, acfh, acdg, bdeg, befh, cegh, (1)*
2^{9-5}_{III}	$5 = 123$, $6 = 124$, $7 = 134$, $8 = 234$, $9 = 1234$	61	72	*abdf, dfgh, cegh, acdg, aefg, abce, befh, bcdh, abcdefghi, acfhi, adehi, cdefi, bdegi, bcfgi, abghi, i*
2^{10-6}_{III}	$5 = 123$, $6 = 124$, $7 = 134$, $8 = 234$, $9 = 1234$, $t_0 = 34$	63	80	*adehi, acfhi, bcfgi, bdegi, abdf, abce, cegh, dfgh, acdgj, aefgj, befhj, bcdhj, cdefij, abcdefghij, abghij, ij*
2^{11-7}_{III}	$5 = 123$, $6 = 124$, $7 = 134$, $8 = 234$, $9 = 1234$, $t_0 = 34$, $t_1 = 24$	75	88	*abdfk, bcdhjk, befhj, abghij, adehi, acfhik, abcdefghijk, bdegik, ijk, cdefij, bcfgi, abce, acdgj, dfgh, ceghk, aefgjk*
2^{12-8}_{III}	$5 = 123$, $6 = 124$, $7 = 134$, $8 = 234$, $9 = 1234$, $t_0 = 34$, $t_1 = 24$, $t_2 = 14$	90	96	*abce, bdegik, adehil, abghij, acfhik, dfgh, abdfkl, abcdefghijkl, ceghkl, acdgjl, bcfgil, befhjl, bcdhjk, cdefij, aefgjk, ijkl*
2^{13-9}_{III}	$5 = 123$, $6 = 124$, $7 = 134$, $8 = 234$, $9 = 1234$, $t_0 = 34$, $t_1 = 24$, $t_2 = 14$, $t_3 = 23$	91	104	*cdefij, befhjl, abdfkl, acfhik, ceghkl, acdgjl, abghij, bdegik, aefgjkm, abcdefghijklm, bcdhjkm, abcem, adehilm, dfghm, bcfgilm, ijklm*
2^{14-10}_{III}	$5 = 123$, $6 = 124$, $7 = 134$, $8 = 234$, $9 = 1234$, $t_0 = 34$, $t_1 = 24$, $t_2 = 14$, $t_3 = 23$, $t_4 = 13$	105	112	*bdegikn, ceghkl, befhjln, cdefij, acdgjln, abcdefghijklmn, bcdhjkm, abghij, acfhikn, abdfkl, adehilm, abcemn, aefgjkm, dfghmn, bcfgilm, ijklmn*
2^{15-11}_{III}	$5 = 123$, $6 = 124$, $7 = 134$, $8 = 234$, $9 = 1234$, $t_0 = 34$, $t_1 = 24$, $t_2 = 14$, $t_3 = 23$, $t_4 = 13$, $t_5 = 12$	120	120	Any order*

*: All run orders have the same LC for these designs, since the designs are saturated. See Cheng, Martin, and Tang (1998) for the theoretical support.

6.4.3. Plackett-Burman designs

Plackett-Burman (P&B) design is another widely-used two-level orthogonal array. For a saturated P&B, or in general any Hadamard design, all possible run orders attain exactly the same LC (see Cheng et al. 1998 and Quinlan and Lin 2015). Thus the LC optimization is only considered for unsaturated P&B designs. An

unsaturated P&B design consists of a subset of factors from the saturated one. Its run order optimization is thoroughly discussed in Quinlan and Lin (2015). Our results fully agree with their results. Moreover, our results obtain some minor extension to their work. Quinlan and Lin (2015) solved the optimal run order for any design that removes no more than 3 columns from a saturated P&B design, while our algorithm works for unsaturated P&B designs of any size. As an illustration, we remove the factor A, B, C, and D from the saturated P&B design in Table 6.6a(a). The remaining design is a 12×7 unsaturated design, as shown in Table 6.6a(b). For this design, our algorithm produces the run order $5 \to 1 \to 12 \to 9 \to 6 \to 8 \to 10 \to 7 \to 4 \to 3 \to 2 \to 11$ (i.e., the 5^{th} run first, the 1^{st} run next, ..., and finally the 11^{th} run). The solution achieves 30 level changes. As a comparison, the average LC is 42 for this design.

Table 6.6: 12-run saturated and unsaturated Plakett-Burman designs

(a) Saturated P&B design

A	B	C	D	E	F	G	H	I	J	K
+	+	−	+	+	+	−	−	−	+	−
−	+	+	−	+	+	+	−	−	−	+
+	−	+	+	−	+	+	+	−	−	−
−	+	−	+	+	−	+	+	+	−	−
−	−	+	−	+	+	−	+	+	+	−
−	−	−	+	−	+	+	−	+	+	+
+	−	−	−	+	−	+	+	−	+	+
+	+	−	−	−	+	−	+	+	−	+
+	+	+	−	−	−	+	−	+	+	−
−	+	+	+	−	−	−	+	−	+	+
+	−	+	+	+	−	−	−	+	−	+
−	−	−	−	−	−	−	−	−	−	−

(b) An unsaturated P&B design

E	F	G	H	I	J	K
+	+	−	−	−	+	−
+	+	+	−	−	−	+
−	+	+	+	−	−	−
+	−	+	+	+	−	−
+	+	−	+	+	+	−
−	+	+	−	+	+	+
+	−	+	+	−	+	+
−	+	−	+	+	−	+
−	−	+	−	+	+	−
−	−	−	+	−	+	+
+	−	−	−	+	−	+
−	−	−	−	−	−	−

It was mentioned that for factorial designs and Plackett-Burman designs, Cheng et al. (1998) and Quinlan and Lin (2015) have provided closed-form constructions of optimal run orders. Their constructions demonstrate *why* those orders have minimum number of level-changes. Compared to these approaches, our algorithm is not theoretically enlightening. But our method can apply to all designs (besides the designs with special structures.)

6.4.4. D-optimal designs

D–optimal design is a wide class of designs which are frequently used in practice. D–optimal designs are orthogonal only for some specific run size n. Their algebraic structures are more complicated, and their run order problem has not been studied in the literature. Nevertheless, the proposed algorithm can apply to D–optimal designs straightforwardly.

Table 6.7: The 6–run *detmax*–design

Run	A	B	C	D	E	F
1	−	+	+	−	−	−
2	+	−	+	−	−	−
3	+	+	−	−	−	−
4	+	+	+	−	+	+
5	+	+	+	+	−	+
6	+	+	+	+	+	−

For illustration, here we present how the proposed works for D–optimal designs under a simple linear model (*detmax*–designs). A table of most two-level *detmax*–designs can be found on the website http://www.indiana.edu/~maxdet/. Table 6.7 shows, the 6–run *detmax*–design, as an example. The algorithm produces an optimal run order $6 \to 5 \to 4 \to 1 \to 3 \to 2$, with 11 level changes.

The proposed algorithm is applied to all *detmax*–designs under 20 runs, and the results are tabulated in Table 6.8. The first column identifies the design, following the notation at the website http://www.indiana.edu/~maxdet/. The minimum LC is given in the second column and a sample optimal run order is provided for practical use.

Table 6.8: Minimum-LC run orders for *detmax*–designs

Design	Minimum LC	Sample optimal run order
$N = 5$	8	Any order*
$N = 6$	11	$6 \to 5 \to 4 \to 1 \to 3 \to 2$
$N = 7$	18	$3 \to 4 \to 7 \to 1 \to 2 \to 6 \to 5$
$N = 9$	30	$5 \to 9 \to 6 \to 1 \to 2 \to 4 \to 7 \to 8 \to 3$
$N = 10$	37	$8 \to 9 \to 10 \to 6 \to 7 \to 4 \to 3 \to 1 \to 5 \to 2$
$N = 11$, R_1**	46	$2 \to 1 \to 3 \to 4 \to 5 \to 9 \to 8 \to 11 \to 10 \to 6 \to 7$
$N = 11$, R_2	46	$2 \to 1 \to 3 \to 4 \to 5 \to 6 \to 7 \to 9 \to 8 \to 10 \to 11$
$N = 11$, R_3	46	$8 \to 9 \to 4 \to 1 \to 5 \to 2 \to 3 \to 11 \to 10 \to 6 \to 7$
$N = 13$	72	Any order*
$N = 14$	79	$8 \to 12 \to 11 \to 10 \to 13 \to 9 \to 14 \to 6 \to 4 \to 2 \to 5 \to 3 \to 7 \to 1$
$N = 15$	90	$4 \to 7 \to 5 \to 6 \to 13 \to 15 \to 14 \to 12 \to 11 \to 10 \to 9 \to 8 \to 2 \to 3 \to 1$
$N = 17$, R_1	130	$3 \to 4 \to 15 \to 7 \to 14 \to 13 \to 17 \to 2 \to 6 \to 12 \to 8 \to 11 \to 9 \to 5 \to 16 \to 10 \to 1$
$N = 17$, R_2	130	$16 \to 4 \to 14 \to 11 \to 9 \to 17 \to 13 \to 3 \to 12 \to 7 \to 2 \to 6 \to 10 \to 8 \to 5 \to 15 \to 1$
$N = 17$, R_3	130	$4 \to 5 \to 3 \to 10 \to 16 \to 8 \to 15 \to 6 \to 13 \to 9 \to 12 \to 11 \to 7 \to 17 \to 2 \to 14 \to 1$
$N = 18$, R_1	137	$15 \to 18 \to 16 \to 13 \to 12 \to 14 \to 10 \to 11 \to 17 \to 8 \to 9 \to 3 \to 6 \to 5 \to 7 \to 4 \to 2 \to 1$
$N = 18$, R_2	137	$12 \to 15 \to 18 \to 17 \to 10 \to 14 \to 13 \to 11 \to 16 \to 2 \to 4 \to 6 \to 9 \to 7 \to 5 \to 8 \to 1 \to 3$
$N = 18$, R_3	137	$16 \to 10 \to 11 \to 14 \to 12 \to 13 \to 15 \to 17 \to 18 \to 5 \to 7 \to 2 \to 4 \to 6 \to 9 \to 8 \to 1 \to 3$
$N = 19$, R_1	152	$13 \to 11 \to 12 \to 6 \to 5 \to 7 \to 1 \to 3 \to 2 \to 4 \to 17 \to 18 \to 19 \to 14 \to 16 \to 15 \to 10 \to 9 \to 8$
$N = 19$, R_2	152	$11 \to 13 \to 12 \to 3 \to 2 \to 4 \to 1 \to 10 \to 9 \to 8 \to 6 \to 5 \to 7 \to 19 \to 17 \to 18 \to 14 \to 15 \to 16$
$N = 19$, R_3	152	$19 \to 18 \to 17 \to 11 \to 12 \to 13 \to 5 \to 6 \to 7 \to 1 \to 4 \to 2 \to 3 \to 15 \to 14 \to 16 \to 9 \to 8 \to 10$

*: For these designs, all run orders achieve the same LC.

**: When there exist non-equivalent Hadamard designs, the notations "$N = 11 : R_1$", "$N = 11, R_2$", and "$N = 11, R_3$" denote three different designs of size 11. Here we follow the notations used at the website http://www.indiana.edu/~maxdet/.

6.5. Theoretical supports

This section is devoted to a theoretical formulation of the run order optimization problem, as well as a rigorous presentation of theoretical results that are used above.

Definition 1. (LC for a run order) Consider an n–run, k–factor experimental design with design matrix $X = (x_{ij})_{1\le i\le n, 1\le j\le k}$. A run order is represented as a sequence $\boldsymbol{R} = \{r_1, r_2, \ldots, r_n\}$, which is a permutation of $\{1, 2, \ldots, n\}$. For a fixed run order $\boldsymbol{R} = \{r_1, r_2, \ldots, r_n\}$, define

$$LC(\boldsymbol{R}) := \sum_{i=1}^{n-1} \sum_{l=1}^{k} I(x_{r_i,l} \neq x_{r_{i+1},l}), \tag{6.1}$$

with $I(\cdot)$ being the indicator function.

Note that LC stands for "the total number of level changes in the run order $\boldsymbol{R}$". By Definition 1, LC can be viewed as a function of the run order $\boldsymbol{R}$. Then it is natural to define the minimum-LC and the corresponding optimal run order for a design.

Definition 2. (Minimum-LC run order) For an n–run design, its minimum-LC is defined as

$$LC_{min} := \min\Big\{LC(\boldsymbol{R}) : \boldsymbol{R} \text{ is any permutation of } \{1, 2, \ldots, n\}\Big\},$$

and the optimal run order (with respect to LC) is $\boldsymbol{R}_{opt} := \arg\min_{\boldsymbol{R}} LC(\boldsymbol{R})$.

This is the theoretical formulation of the run order optimization problem, and it can be solved via the proposed algorithm. The input of algorithm is the design matrix X, and the output is the permutation (vector) $\boldsymbol{R}_{opt}$. Throughout the process

of our algorithm, a crucial middle-product is the *adjacency matrix of distance graph*, and its general definition is as follows.

Definition 3. (Adjacency matrix of distance graph) Consider an n–run design X, and the distances between runs are pre-specified. Then its adjacency matrix of distance graph is an n by n matrix

$$A = (d_{ij})_{1 \le i,j \le n},$$

where d_{ij} is the distance between run i and run j, and $d_{ii} = 0$.
In particular, when the pre-specified distance is the number of level changes,

$$A = \left(\sum_{l=1}^{k} I(x_{il} \neq x_{jl}) \right)_{1 \le i,j \le n}.$$

In general, the adjacency matrix A for LC can be obtained by the definition. This can be cumbersome in some cases. The following Theorem provides an easy construction of the matrix A. This makes Step-1 of the proposed algorithm computationally efficient.

Theorem 11. *For a two-level design with n runs and k factors, denote X as its $n \times k$ design matrix. Then the adjacency matrix of its distance graph is*

$$A = (kE - XX^T)/2, \tag{6.2}$$

where E is the $n \times n$ matrix with all entries being 1.

For example, if the design is 2^4 factorial, then X^T is the matrix in Table 6.2. Plug

this X^T into $A = (kE - XX^T)/2$, one immediately obtains the adjacency matrix in Table 6.3.

Once the adjacency matrix A is obtained, one can follow the proposed algorithm to modify A, input the modified adjacency matrix into the TSP solver, and then obtain an optimal run order. In practice, it is desirable to compare the optimal LC obtained with the *average-LC* among all possible run orders. So here we define the LC-distribution, and the average-LC.

Definition 4. Consider any $n-$run design. Define a sample space of all possible run orders

$$\Omega := \Big\{ \boldsymbol{R} : \boldsymbol{R} \text{ is any permutation of } \{1, 2, \ldots, n\} \Big\},$$

with all events occuring with equal probability. Then LC can be viewed as random variable on Ω, and the distribution of this random variable is called the LC-distribution. The expectation of this distribution is called the average-LC, denoted by $\mathrm{E}(LC)$.

From Definition 4 the average-LC is

$$\mathrm{E}(LC) = \frac{1}{n!} \sum_{\boldsymbol{R}} LC(\boldsymbol{R}),$$

where $\boldsymbol{R}$ takes every permutation. For a small number of runs $E(LC)$ can be evaluated by enumerating all permutations. However, we have the following theorem for any run size n. This is especially useful for large n.

Theorem 12. *For an n-run design and any distance, denote the adjacency matrix of distance graph as A, then the average-LC can be calculated as the following function*

of A:

$$\mathrm{E}(LC) = \boldsymbol{e}^T A\boldsymbol{e}/n, \tag{6.3}$$

where $\boldsymbol{e}$ *is the* $n-$*dimensional vector with all entries being 1.*

As an example, if $n = 16$ and A is the adjacency matrix in Table 6.2, then $\boldsymbol{e}^T A\boldsymbol{e}/n$ equals 32. This is the average LC for the 2^4 design. In fact, for any balanced orthogonal design, the average-LC only depends on the size of design, i.e., the number of runs and factors. We could state the result for the 2−level case as following.

Corollary 2. *For any* 2*−level balanced orthogonal design with* k *factors and* n *runs,* $\mathrm{E}(LC) = kn/2$.

By this corollary we can immediately see the average-LC for a wide variety of 2−level designs.

6.6. Further applications

6.6.1. Minimum-LC run orders for higher-level designs

Besides the two−level designs discussed in the last section, the proposed algorithm is straightforwardly applicable to other designs. For illustration, we show how this works for Box-Behnken designs. Box-Behnken design is a class of 3−level designs that are widely used in response surface methodology and industrial optimization. Table 6.9 displays a 15−run Box-Behnken design with factors A, B, C, each at three levels "−", "0" and "+".

Assuming the LC is equally costly among all level pairs $(- \leftrightarrow 0)$, $(0 \leftrightarrow +)$, and $(+ \leftrightarrow -)$. The algorithm produces a run order $1 \to 2 \to 3 \to 4 \to 8 \to 9 \to 7 \to 6 \to$

Table 6.9: A Box-Behnken Design

Run	A	B	C
1	−	−	0
2	+	−	0
3	−	+	0
4	+	+	0
5	0	0	0
6	−	0	−
7	+	0	−
8	−	0	+
9	+	0	+
10	0	0	0
11	0	−	−
12	0	+	−
13	0	−	+
14	0	+	+
15	0	0	0

$12 \to 11 \to 13 \to 14 \to 15 \to 5 \to 10$, with its LC being 15. The average LC is 28.8, and thus the optimal order reduces the LC by half of the average. Note that if the costs for different pairs of level-change are not the same, all we need to do is to modify the distance between any two runs.

Joseph (2009) studied the optimal run order for the mixed-level $L_{18}(2^1 \times 3^7)$ design in Schrijver (1998b). Note that this is the only illustrative example given in Joseph's paper. He has pre-specified the cost of level-change for each factor, then applied TSP to optimize the run order. However, his setup is very different from ours here: in addition to the specified costs from changing levels, there is a pre-specified initial run (denoted by Run "0"). Note that Run "0" is not in the $L_{18}(2^1 \times 3^7)$ design. Moreover, after all 18 runs were conducted, one must return to the initial run "0". In other words, the run order problem given in Joseph's is to start with a fixed Run "0", run all experiments, and then end with again a replication of Run "0", a total of $18 + 2 (= 20)$ runs. This is rather unusual in practice. Under this special setup,

the run order optimization is exactly equivalent to TSP. A TSP solver obtained the optimal run order $0 \to 2 \to 1 \to 4 \to 9 \to 17 \to 11 \to 10 \to 16 \to 14 \to 15 \to 13 \to 12 \to 18 \to 7 \to 8 \to 3 \to 6 \to 5 \to 0$ (with a total cost of 40 under his cost definition).

However, there is no such initial/end Run "0" in reality (for example, all the designs investigated in Section 4). Consequently, Joseph (2009) approach does not work here. On the other hand, without such an initial/end run, the TSP cannot be applied directly. Step 2 in our algorithm is critical. It creates a dummy node (called it a "virtual node"), which has zero distance to all design nodes. With such a dummy node, the run order problem becomes a standard TSP. The output from TSP solver is essentially a loop. As a result, the node after the dummy node is the initial run, while the node before the dummy node is the end run. The dummy node itself will not be performed at all. This way, our result produces a sequence of an optimal run order (with all runs involved).

With the identical cost structure given in Joseph (2009), our algorithm has been implemented in two ways: (a) Suppose there is indeed an initial run–the same Run "0" in Joseph (2009), but there is no need to return to Run "0" at the end. In this case, TSP cannot apply directly (thus Joseph's method will also fail). Our algorithm can provide an optimal run order, which is $0 \to 5 \to 2 \to 7 \to 9 \to 4 \to 1 \to 3 \to 6 \to 8 \to 16 \to 10 \to 11 \to 17 \to 18 \to 12 \to 13 \to 15 \to 14$ (with a total cost of 37). (b) Suppose there is no initial Run "0", i.e., the experimenter simply hopes to conduct the 18 runs in the L_{18} design. In this case, TSP cannot apply directly either. Our algorithm provides an optimal run order of $12 \to 18 \to 16 \to 15 \to 14 \to 13 \to 11 \to 17 \to 9 \to 10 \to 8 \to 1 \to 6 \to 3 \to 2 \to 5 \to 7 \to 4$. (with a total cost of 35.5).

Note that the proposed algorithm is also applicable to all kinds of designs, such as, mixed-level designs, central composite designs, balanced incomplete block designs, as well as definitive screening designs.

6.6.2. Optimal run orders with other distance measures

In the preceding sections, we focus on the LC distance. In some practice, the experimenter is concerned not only about the LC *number*, but about the *cost* of resetting factors. The cost can be either time or monetary consumption, e.g., the waiting time to replace a component, or the energy expense of reheating/cooling (see Joiner and Campbell 1976).

Consider the Box-Behnken design in Table 6.8. Suppose it takes 60 seconds to reset factor A, 20 seconds to reset factor B, and 5 seconds to reset factor C. The goal is to find an optimal run order which minimizes the total time in resetting levels. To apply the proposed algorithm, the distance between runs is defined as $60 \times I(A \text{ changes}) + 20 \times I(B \text{ changes}) + 5 \times I(C \text{ changes})$, where $I(\cdot)$ is the indicator function. For instance, the distance between Run 2 and Run 3 is $60 + 20 = 80$, and the distance between Run 10 and Run 11 is $20 + 5 = 25$.

The proposed algorithm then produces an optimal run order $1 \to 3 \to 6 \to 8 \to 9 \to 7 \to 2 \to 4 \to 14 \to 12 \to 11 \to 13 \to 15 \to 5 \to 10$. With this run order it takes 280 seconds in total to reset factors. As a comparison, the average time cost among all possible run oreders is 816 seconds (about three times the optimal 280 seconds).

In general, the distance criterion can be defined in a flexible way. When there are multiple criteria for the run order, we may also weight the distances to define a compound distance.

Chapter 7

Conclusion and future work

7.1. Conclusions

This dissertation has established a number of new theories and design construction methods in three different design problems: design of order-of-addition experiments, optimal design for estimating error variance, and construction of optimal run orders. The main conclusions of each chapter are highlighted and interpreted here.

- **Chapters 2 and 3:** These two chapters are motivated by two fundamental questions: (1) When is an order-of-addition design the most efficient (i.e., optimal)? (2) What is the efficiency of an optimal order-of-addition design?

  Instead of answering the question under different models one by one, Chapter 2 gives a generic answer to question (1) for any existing order-of-addition model in the literature (as far as we know). It is shown that under all these models, an order-of-addition design is D-optimal (but not necessarily optimal under other criteria) if and only if its moment matrix is the same as that of full design.

  Chapter 3 focuses on a specific type of model: pairwise order models with possible tapering. Under pairwise order model, comprehensive answers are

given to both questions (1) and (2). An order-of-addition design is optimal under each of the commonly used D-, A-, E-, M.S.- criteria, if and only if its moment matrix is the same as that of full design. Furthermore, the efficiency of optimal design is explicitly derived, which provides benchmark for assessing the property of any other design. We also give closed-form construction for a class of optimal designs for any number of components m, which shows that the optimality condition is attainable. These closed-form designs possess not only the optimality under pairwise order model, but also the robustness to other models, and are thus useful (when m is small).

- **Chapter 4:** Construction of order-of-addition design is difficult when m is large, and a new algorithm called "Swap-r" is proposed here to resolve this issue. The Swap-r algorithm is widely applicable to any order-of-addition linear model and almost any number of runs. The algorithm is computationally favorable: it produces designs with m up to 50 in reasonable time. The algorithm employs multiple techniques to escape local optima and results in quite efficient designs. Many features of this algorithm are extendable to other design problems.

- **Chapter 5:** "Saturated design appended by center points" is a conventional choice of small screening design. We propose an alternative screening design: partial columns of a Hadamard matrix. Under some assumptions, the proposed alternative is shown to be optimal for estimating error variance and more favorable for significance tests than the conventional choice. These conclusions are further extended to the "follow-up design" problem in the later half of Chapter 5, with a bunch of new theoretical results. These results manifest the subtle impact of designs on the estimation of error variance, which was rarely

explored in the literature.

- **Chapter 6:** We propose an algorithm to optimize the run-order in design of experiments, by converting the run-order problem into a Traveling Salesman Problem (TSP) and applying a modern TSP solver. The proposed algorithm efficiently produces a bunch of minimum-level-change run orders that are new in the literature. This helps reducing economic expense/time consumption in many experiments. The proposed algorithm is applicable to any type of designs, while existing methods are limited to designs with specific structures.

7.2. Future work

This section summarizes some on-going or potential work following my dissertation. A large area of future work is in the real application of order-of-addition design. I will first illustrate this topic in details, and then list other future topics.

Real applications of order-of-addition design

As mentioned in Section 3.2, we ultimately desire some order-of-addition designs that are (nearly-)optimal under pairwise-order model and are meanwhile robust to other models. Of course, for practical use, designs should have small number of runs. The optimal pairwise order designs obtained in Chapter 4 are suitable candidates, except that they are not necessarily robust to other models. As such, a future work is to apply some multi-objective optimization or constrained optimization techniques, for obtaining robust designs. Thus obtained designs should be useful in many real chemical experiments. We plan to collaborate with chemists and evaluate whether one can efficiently learn the order effects via these designs.

Another topic regarding the real application is the inference of optimal order(s).

In the order-of-addition experiments with large number of components m, it is difficult to infer the optimal order(s) even if the order effects are efficiently learned. This is because inferring the optimal order(s) typically requires calculating the estimated responses of all the $m!$ orders, which is computationally unaffordable. This blocks the application of order-of-addition experiments in many areas, say, job scheduling (see, e.g., Pinedo 2016). We have proposed a way of resolving this computational issue, under the pairwise order model. The idea is to treat different components as nodes in a graph, and formulate (significant) pairwise order effects as directional edges on the graph. Then, the optimal orders should be among the solutions of topological sorting (see, e.g., Kahn 1962) on this graph; we thus need to search optimal order(s) only in a small portion of orders. As such, if the pairwise order model explains the majority of response variation for a job scheduling problem, then with an efficient pairwise order design, one should be able solve the (nearly-)optimal job orders in short time. This is appealing in view that many scheduling problems are NP-hard and thus cannot be efficiently solved by existing algorithms in reasonable time. We are currently testing the proposed methodology on some scheduling problems.

Other research topics

- High-way order-of-addition modeling and designing. Section 2.3 introduced the pairwise and triplet order models. How about 4-way, or higher-way models? Mee (2019) proposed a protocol of building high-way models, and its high-level ideas have been mentioned in Section 2.6. However, the explicit forms of high-way models have not been derived yet. A future work is to derive such explicit form and construct designs accordingly. In addition, the super-saturated designs (Lin and Draper 1993) will be of interest under triplet or higher way models, as the number of possible effects tremendously increase in

m and therefore, even a saturated design would be too expensive under these models.

- Extension of Swap-r algorithm into other design problems. As mentioned in Section 4.7, many features of the Swap-r algorithm are quite generic. It can be applied to the construction of large-scale factorial designs, after a small amount of modifications. It will a meaningful work to implement the modified Swap-r and compare its performance with existing methods for the construction of D-efficient factorial designs.

- Extension of the optimal design for estimating error variance. Chapter 5 considers the optimal design under linear models (for which the two-level designs are appropriate). In the presence of, say, quadratic effects, three-level designs are needed. It will be useful to extend the results in Chapter 5 to three- or higher-level cases. The consideration of interactions is also meaningful. In the presence of (sparse) interactions, there might also be optimal designs and corresponding analysis methods to separate the interactions with the pure error, which result in reliable estimates of error variance and significance tests of main effects.

- Solving optimization problems in DOE using modern combinatorial optimization. An important spirit of Chapter 6 is to utilize the modern developments on graphical algorithms (explicitly, the TSP solver) in design construction. People used to prefer constructing designs in closed form according to designs' algebraic structure, because at that time, perhaps decades ago, it was computationally unaffordable to construct many designs using some generic devices (such as TSP solver). Yet now, it becomes possible to solve many NP-hard problems efficiently. We hope to construct more designs using updated tools

in combinatorial optimization.

Bibliography

T. W. Anderson. *An introduction to multivariate statistical analysis*, volume 2. Wiley New York, 1958.

C. M. Anderson-Cook and L. Lu. Discussion on order-of-addition experiments: A review and some new thoughts. *Quality Engineering*, 31(1):64–68, 2019.

D. L. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook. Concorde: a code for solving traveling salesman problems, 2003. `http://www.math.uwaterloo.ca/tsp/concorde.html`.

D. L. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook. *The Traveling Salesman Problem: a Computational Study*. Princeton University Press, 2011.

A. C. Atkinson and R. D. Cook. D-optimum designs for heteroscedastic linear models. *Journal of the American Statistical Association*, 90(429):204–212, 1995.

Z. Bai and J. W. Silverstein. *Spectral analysis of large dimensional random matrices*, volume 20. Springer, 2010.

R. R. Barton. *Graphical methods for the design of experiments*, volume 143. Springer Science & Business Media, 2012.

K. H.V. Booth and D. R. Cox. Some systematic supersaturated designs. *Technometrics*, 4(4):489–495, 1962.

G. E. P. Box and R. D. Meyer. An analysis for unreplicated fractional factorials. *Technometrics*, 28(1):11–18, 1986.

G. E.P. Box and N. R. Draper. *Empirical model-building and response surfaces*, volume 424. Wiley New York, 1987.

K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *Statistical Science*, pages 273–304, 1995.

C. S. Cheng. Innovation and intellectual property rights. In S. Ghosh, editor, *Construction of Run Orders of Factorial Designs*, pages 423–439. Marcel Dekker, New York, 1990.

C. S. Cheng and M. Jacroux. The construction of trend-free run orders of two-level factorial designs. *Journal of the American Statistical Association*, 83(404): 1152–1158, 1988.

C. S. Cheng and D. M. Steinberg. Trend robust two-level factorial designs. *Biometrika*, 78(2):325–336, 1991.

C. S. Cheng, R. J. Martin, and B. Tang. Two-level factorial designs with extreme numbers of level changes. *The Annals of Statistics*, 26(4):1522–1539, 1998.

H. Chernoff. Locally optimal designs for estimating parameters. *The Annals of Mathematical Statistics*, pages 586–602, 1953.

D. C. Coster. Tables of minimum cost, linear trend-free run sequences for two-and three-level fractional factorial designs. *Computational Statistics and Data Analysis*, 16:325–336, 1993.

D. C. Coster and C.S. Cheng. Tables of minimum cost, linear trend-free run sequences for two-and three-level fractional factorial designs. *The Annals of Statistics*, 16(3): 1188–1205, 1988.

D. R. Cox. Some systematic experimental designs. *Biometrika*, 38(3/4):312–323, 1951.

C. Daniel. Use of half-normal plots in interpreting factorial two-level experiments. *Technometrics*, 1(4):311–341, 1959.

C. Daniel. Half-normal plots. In S. Kotz and N. L. Johnson, editors, *Encyclopedia of Statistical Sciences*, volume 3. John Wiley, New York, 1983.

G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America*, 2(4):393–410, 1954.

L. T. DeCarlo. On the meaning and use of kurtosis. *Psychological methods*, 2(3): 292, 1997.

H. Dette and W. K. Wong. Bayesian d-optimal designs on a fixed number of design points for heteroscedastic polynomial models. *Biometrika*, 85(4):869–882, 1998.

A. W. Dickinson. Some run orders requiring a minimum number of factor level changes for the 24 and 25 main effect plans. *Technometrics*, 16(1):31–37, 1974.

X. Ding, K. Matsuo, L. Xu, J. Yang, and L. Zheng. Optimized combinations of bortezomib, camptothecin, and doxorubicin show increased efficacy and reduced toxicity in treating oral cancer. *Anti-cancer drugs*, 26(5):547–554, 2015.

F. Dong. On the identification of active contrasts in unreplicated fractional factorials. *Statistica Sinica*, pages 209–217, 1993.

N. R. Draper and D. M. Stoneman. Factor changes and linear trends in eight-run two-level factorial designs. *Technometrics*, 10(2):31–37, 1968.

J. A. Eccleston and A. Hedayat. On the theory of connected designs: characterization and optimality. *The Annals of Statistics*, 2(6):1238–1255, 1974.

S. Ehrenfeld. On the efficiency of experimental designs. *The Annals of Mathematical Statistics*, 26(2):247–255, 1955.

V. V. Fedorov. *Theory of Optimal Experiments*. Elsevier, 1972.

T. Fuleki and F. J. Francis. Quantitative methods for anthocyanins. *Journal of Food Science*, 33(3):266–274, 1968.

J. Ganju and J. M. Lucas. Bias in test statistics when restrictions in randomization are caused by factors. *Communications in Statistics: Theory and Methods*, 26(1): 47–63, 1997.

S. G. Gilmour and L. A. Trinca. Optimum design of experiments for statistical inference. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 61(3):345–401, 2012.

D. Henry, L. L. Y. Lim, L. A. G. Rodriguez, S. P. Gutthann, J. L. Carson, M. Griffin, R. Savage, R. Logan, Y. Moride, and C. Hawkey. Variability in risk of gastrointestinal complications with individual non-steroidal anti-inflammatory drugs: results of a collaborative meta-analysis. *Bmj*, 312(7046):1563–1566, 1996.

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

H. H. Hoos and T. Stützle. On the empirical scaling of run-time for finding optimal solutions to the travelling salesman problem. *European Journal of Operational Research*, 238:87–94, 2014.

J. C. Hsu. Constrained simultaneous confidence intervals for multiple comparisons with the best. *The Annals of Statistics*, 12(3):1136–1144, 1984.

P. D. Hurley. The conservative nature of the effect sparsity assumption for saturated fractional factorial experiments. *Quality Engineering*, 7(4):657–671, 1995.

M. E. Johnson and C. J. Nachtsheim. Some guidelines for constructing exact d-optimal designs on convex design spaces. *Technometrics*, 25(3):271–277, 1983.

S. M. Johnson. Generation of permutations by adjacent transposition. *Mathematics of computation*, 17(83):282–285, 1963.

B. L. Joiner and C. Campbell. Designing experiments when run order is important. *Technometrics*, 18(3):249–259, 1976.

B. Jones and D. C. Montgomery. Partial replication of small two-level factorial designs. *Quality Engineering*, 29(2):190–195, 2017.

B. Jones and C. J. Nachtsheim. Split-plot designs: What, why, and how. *Journal of Quality Technology*, 41:387–393, 2009.

B. Jones, D. K.J. Lin, and C. J. Nachtsheim. Bayesian d-optimal supersaturated designs. *Journal of Statistical Planning and Inference*, 138(1):86–92, 2008.

R. V. Joseph. Experimental sequence: a decision strategy. *Quality Engineering*, 12 (3):387–393, 2009.

L. S. Jourdain, C. Schmitt, M. E. Leser, B. S. Murray, and E. Dickinson. Mixed layers of sodium caseinate+ dextran sulfate: influence of order of addition to oil-water interface. *Langmuir*, 25(17):10026–10037, 2009.

A. B. Kahn. Topological sorting of large networks. *Communications of the ACM*, 5 (11):558–562, 1962.

M. Karim, K. McCormick, and C. T. Kappagoda. Effects of cocoa extracts on endothelium-dependent relaxation. *The Journal of Nutrition*, 130(8):2105S–2108S, 2000.

R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Springer, 1972.

J. Kiefer. General equivalence theory for optimum designs (approximate theory). *The Annals of Statistics*, 2(5):849–879, 1974.

J. Kiefer. Construction and optimality of generalized youden designs. In J. N. Srivastava, editor, *A Survey of Sta- tistical Design and Linear Models*. North Holland, Amsterdam, 1975.

J. Kiefer and J. Wolfowitz. Optimum designs in regression problems. *The Annals of Mathematical Statistics*, 30(2):271–294, 1959.

J. Kiefer and J. Wolfowitz. The equivalence of two extremum problems. *Canadian Journal of Mathematics*, 12(363-366):234, 1960.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

D. E. Knuth and J. L. Szwarcfiter. A structured program to generate all topological sorting arrangements. *Information Processing Letters*, 2(6):153–157, 1974.

W. F. Kuhfeld and R. D. Tobias. Large factorial designs for product engineering and marketing research applications. *Technometrics*, 47(2):132–141, 2005.

Ju H. L. and J. M. Lucas. l^k factorial experiments with hard-to-change and easy-to-change factors. *Journal of Quality Technology*, 34(4):411–421, 2002.

R. Lekivetz, R. Sitter, D. Bingham, M. S. Hamada, L. M. Moore, and J. R. Wendelberger. On algorithms for obtaining orthogonal and near-orthogonal arrays for main-effects screening. *Journal of Quality Technology*, 47(1):2, 2015.

R. V. Lenth. Quick and Easy Analysis of Unreplicated Factorials. *Technometrics*, 31(4):469–473, 1989.

R. D. Leonard and D. J. Edwards. Bayesian d-optimal screening experiments with partial replication. *Computational Statistics & Data Analysis*, 2017.

D. K.J. Lin. Generating systematic supersaturated designs. *Technometrics*, 37(2): 213–225, 1995.

D. K.J. Lin and N. R. Draper. Projection properties of plackett and burrnan designs. *Technometrics*, 34(4):423–428, 1992.

D. K.J. Lin and N. R. Draper. Generating alias relationships for two-level plackett and burman designs. *Computational statistics & data analysis*, 15(2):147–157, 1993.

D. K.J. Lin and J. Peng. Rejoinder. *Quality Engineering*, 31(1):69–72, 2019a.

D. K.J. Lin and J. Peng. Order-of-addition experiments: A review and some new thoughts. *Quality Engineering*, 31(1):49–59, 2019b.

D. K.J. Lin and Lam A. W. Connections between two-level factorials and venn diagrams. *The American Statistician*, 51(1):49–51, 1997.

R. W. Mee. Order of addition modeling. Submitted for publication, 2019.

R. K. Meyer and C. J. Nachtsheim. The coordinate-exchange algorithm for constructing exact optimal experimental designs. *Technometrics*, 37(1):60–69, 1995.

A. Miller. The analysis of unreplicated factorial experiments using all possible comparisons. *Technometrics*, 47(1):51–63, 2005.

D. C. Montgomery. *Design and Analysis of Experiments*. John Wiley & Sons, 2008.

R. Mukerjee and C.F. J. Wu. *A modern theory of factorial design*. Springer Science & Business Media, 2007.

M. Nawaz, E. E. Enscore Jr, and I. Ham. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, 11(1):91–95, 1983.

B. L. Nelson. Robust multiple comparisons under common random numbers. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 3(3):225–243, 1993.

B. L. Nelson and F. J. Matejcik. Using common random numbers for indifference-zone selection and multiple comparisons in simulation. *Management Science*, 41 (12):1935–1945, 1995.

R. L. Obenchain. Classical f-tests and confidence regions for ridge regression. *Technometrics*, 19(4):429–439, 1977.

G. J. Olsen, H. Matsuda, R. Hagstrom, and R. Overbeek. Fastdnaml: a tool for construction of phylogenetic trees of dna sequences using maximum likelihood. *Computer Applications in the Biosciences: CABIOS*, 10(1):41–48, 1994.

P. Y. Papalambros and D. J. Wilde. *Principles of optimal design: modeling and computation*. Cambridge university press, 2000.

A. Parikh, R. Mantravadi, D. Kozhevnikov, M. A. Roche, Y. Ye, L. J. Owen, J. L. Puglisi, J. J. Abramson, and G. Salama. Ranolazine stabilizes cardiac ryanodine receptors: a novel mechanism for the suppression of early afterdepolarization and torsades de pointes in long qt type 2. *Heart Rhythm*, 9(6):953–960, 2012.

J. Peng and D. K. J. Lin. Small screening design when the overall variance is unknown. *Journal of Statistical Planning and Inference*, 2019a. doi: 10.1016/j.jspi.2019.04.011.

J. Peng and D. K.J. Lin. Construction of optimal run order in design of experiments. *Journal of Quality Technology*, 51(2):159–170, 2019b.

J. Peng, R. Mukerjee, and D. K.J. Lin. Design of order-of-addition experiments. *Biometrika*, 106(3):683–694, 2019.

M. L. Pinedo. *Scheduling: theory, algorithms, and systems*. Springer, 2016.

R. L. Plackett and J. P. Burman. The design of optimum multifactorial experiments. *Biometrika*, 33(4):305–325, 1946.

F. Pukelsheim. *Optimal design of experiments*, volume 50. siam, 1993.

F. Pukelsheim. *Optimal Design of Experiments*. Society for Industrial and Applied Mathematics, 2006. doi: 10.1137/1.9780898719109.

F. Pukelsheim and J. L. Rosenberger. Experimental designs for model discrimination. *Journal of the American Statistical Association*, 88(422):642–649, 1993.

K. R. Quinlan and D. K.J. Lin. Run order considerations for plackett and burman designs. *Journal of Statistical Planning and Inference*, 165:56–62, 2015.

D. Raghavarao. *Constructions and combinatorial problems in design of experiments*. Wiley, 1971.

M. Rajaonarivony, C. Vauthier, G. Couarraze, F. Puisieux, and P. Couvreur. Development of a new drug carrier made from alginate. *Journal of Pharmaceutical Sciences*, 82(9):912–917, 1993.

D. Rodbard, H.J. Ruder, J. Vaitukaitis, and H.S. Jacobs. Mathematical analysis of kinetics of radioligand assays: Improved sensitivity obtained by delayed addition of labeled ligand. *The Journal of Clinical Endocrinology & Metabolism*, 33(2): 343–355, 1971.

R. Ruiz and T. Stützle. A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research*, 177(3):2033–2049, 2007.

P. Ryberg. Development of a mild and robust method for large-scale palladium-catalysed cyanation of aryl bromides: Importance of the order of addition. *Organic Process Research & Development*, 12(3):540–543, 2008.

SAS Institute Inc (SAS). Sas/qc 13.2 users guide: The optex procedure. Retrieved November 8, 2017 from https://support.sas.com/documentation/onlinedoc/qc/132/optex.pdf, 2017.

A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, 1998a.

A. Schrijver. *System of Experimental Design*. UNIPUB/Kraus International Publications, 1998b.

G. A. Seber and A. J. Lee. *Linear regression analysis*, volume 936. John Wiley & Sons, 2012.

A. Shinohara and T. Ogawa. Stimulation by rad52 of yeast rad51-mediated recombination. *Nature*, 391(6665):404–407, 1998.

S. D. Silvey. *Optimal design: an introduction to the theory for parameter estimation*, volume 1. Springer Science & Business Media, 2013.

K. Smith. On the standard deviations of adjusted and interpolated values of an observed polynomial function and its constants and the guidance they give towards a proper choice of the distribution of observations. *Biometrika*, 12(1/2):1–85, 1918.

Y. Song, S. Zhu, S. Xiang, X. Zhao, J. Zhang, H. Zhang, Y. Fu, and B. Yang. Investigation into the fluorescence quenching behaviors and applications of carbon dots. *Nanoscale*, 6(9):4676–4682, 2014.

C. A. Stewart, D. Hart, D. K. Berry, G. J. Olsen, E. A. Wernert, and W. Fischer. Parallel implementation and performance of fastdnaml-a program for maximum likelihood phylogenetic inference. In *Supercomputing, ACM/IEEE 2001 Conference*, pages 32–32. IEEE, 2001.

W. E. Strawderman. Minimax adaptive generalized ridge regression estimators. *Journal of the American Statistical Association*, 73(363):623–627, 1978.

T. Sugimoto, K. Sakata, and A. Muramatsu. Formation mechanism of monodisperse pseudocubic α-fe2o3 particles from condensed ferric hydroxide gel. *Journal of colloid and interface science*, 159(2):372–382, 1993.

C. M. Theobald. Generalizations of mean square error applied to ridge regression. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 103–106, 1974.

B. Torsney and R. Martín-Martín. Multiplicative algorithms for computing optimum designs. *Journal of Statistical Planning and Inference*, 139(12):3947–3961, 2009.

W. Townsend. The single machine problem with quadratic penalty function of completion times: a branch-and-bound solution. *Management Science*, 24(5):530–534, 1978.

S. F. Tsai and C. T. Liao. Selection of partial replication on two-level orthogonal arrays. *Canadian Journal of Statistics*, 42(1):168–183, 2014.

R. C. Van Nostrand. Design of experiments where the order of addition is important. In *ASA Proceedings of the Section on Physical and Engineering Sciences*, pages 155–160, Alexandria, Virginia, 1995. American Statistical Association.

J. G. Voelkel. The design of order-of-addition experiments. *Journal of Quality Technology*, pages 1–12, 2019.

A. Wald. On the efficient design of statistical investigations. *The Annals of Mathematical Statistics*, 14(2):134–140, 1943.

M. P. Wand and M. C. Jones. *Kernel smoothing*. Crc Press, 1994.

P. C. Wang and H. W. Jan. Designing two-level factorial experiments using orthogonal arrays when the run order is important. *The Statistician*, 44(3):379–388, 1995.

D. F. Webb, J. M. Lucas, and J. J. Borkowski. Designing two-level factorial experiments using orthogonal arrays when the run order is important. *The Statistician*, 36(1):1–11, 2004.

W. J. Welch. Branch-and-bound search for experimental designs based on d–optimality and other criteria. *Technometrics*, 24(1):41–48, 1982.

C.F. J. Wu and M. S. Hamada. *Experiments: Planning, Analysis, and Optimization*, volume 552. John Wiley & Sons, 2011.

H. Xu. Minimum moment aberration for nonregular designs and supersaturated designs. *Statistica Sinica*, 13(3):691–708, 2003.

H. Xu. Algorithmic construction of efficient fractional factorial designs with large run sizes. *Technometrics*, 51(3):262–277, 2009.

J. Yang, F. Sun, and H. Xu. Component orthogonal arrays for order-of-addition experiments. Submitted for publication, 2019.

K. Q. Ye, M. Hamada, and C.F. J. Wu. A step-down lenth method for analyzing unreplicated factorial designs. *Journal of Quality Technology*, 33(2):140, 2001.

Y. Zhao, D. K.J. Lin, and M. Liu. Design of order of addition experiment. Submitted for publication, 2019.

Vita

Jiayu Peng

Education

- **Ph.D., Statistics** The Pennsylvania State University, University Park, PA, 2019

  Dissertation: New Developments in Design of Experiments

  Advisor: Dr. Dennis K.J. Lin

- **B.S., Mathematics** Tsinghua University, Beijing, China, 2012

Publications

- **Peng, Jiayu**, and Dennis K.J. Lin. "Construction of optimal run order in design of experiments." *Journal of Quality Technology* 51.2 (2019): 159-170.

- **Peng, Jiayu**, and Dennis K.J. Lin. "Small screening design when the overall variance is unknown." *Journal of Statistical Planning and Inference* (2019), in press, doi: 10.1016/j.jspi.2019.04.011.

- **Peng, Jiayu**, Rahul Mukerjee, and Dennis K.J. Lin. "Design of order-of-addition experiments." *Biometrika* 106.3 (2019): 683-694.

- Lin, Dennis K.J., and **Jiayu Peng**. "Order-of-addition experiments: A review and some new thoughts (with discussion and a rejoinder by the authors)." *Quanlity Engineering* 31.1 (2019): 49-59.

- **Peng, Jiayu**, Russel R. Barton, and Dennis K.J. Lin. "Fast Swap-r Algorithm for Construction of Efficient Order-of-addition Designs". Submitted article.

- Meng, Huicui, Zhaoyong Ba, Yujin Lee, **Jiayu Peng**, Junli Lin, Jennifer A. Fleming, Emily J. Furumoto, Robert F. Roberts, Penny M. Kris-Etherton, and Connie J. Rogers. "Consumption of Bifidobacterium animalis subsp. lactis BB-12 in yogurt reduced expression of TLR-2 on peripheral blood-derived monocytes and pro-inflammatory cytokine secretion in young adults." *European journal of nutrition* 56.2 (2017): 649-661.