The Pennsylvania State University
The Graduate School

CONTRIBUTIONS TO MIXTURE EXPERIMENTS WHEN ORDER OF ADDITION IS IMPORTANT

A Dissertation in
Statistics
by
Nicholas Rios

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2022

The dissertation of Nicholas Rios was reviewed and approved by the following:

Lingzhou Xue
Associate Professor of Statistics
Dissertation Advisor
Chair of Committee

Murali Haran
Professor and Department Head of Statistics

Yanyuan Ma
Professor of Statistics

Hui Yang
Professor of Industrial and Manufacturing Engineering

Dennis Lin
Distinguished Professor Emeritus and Head of the Department of Statistics, Purdue University, and Special Member

Ephraim Mont Hanks
Associate Professor of Statistics
Chair of Graduate Studies

Abstract

In a mixture experiment, m components are mixed to produce a response. This is a classical problem in chemical engineering, pharmaceutical, and food science fields. However, existing literature on mixture designs ignores the order of addition of the mixture components. We consider the Order-of-Addition (OofA) mixture experiment, where the response depends on the order of addition of the m components, as well as their mixture proportions. The overall goal of this experiment is to identify the addition order and mixture proportions that produce an optimal response. Full OofA Mixture designs are created which ensure orthogonality between mixture model terms and OofA effects. These designs support models with (1) typical mixture parameters, (2) order-of-addition effects, and (3) interactions between mixture and order terms. Simulations show that if interactions exist, then the optimal mixture proportions identified by traditional models may be misleading. While the full OofA Mixture designs are useful, the number of runs they require increases rapidly as m increases. We propose the use of computer algorithms to search of a subset of runs from the full OofA Mixture design that maximize an optimality criterion. In particular, a Threshold Accepting (TA) algorithm is proposed to find optimal subsets of the full design. Finally, we investigate the scenario where the set of all possible orders in a chemical experiment is restricted to those permissible on an undirected graph, such as a chemical reaction network. Sufficient conditions for estimability of the popular pariwise ordering model are derived for this scenario. Depth-First Search (DFS) is used to enumerate the set of all possible Hamiltonian paths on a graph. A fractional Depth-First Search (DFS) approach is proposed to find highly efficient fractions of the full DFS design, which are shown to have robust efficiencies under different random graphical models.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank my family for their continued love and support throughout my time at Penn State. My husband has been my most steadfast supporter, giving me the inspiration and drive to pursue my dreams. My parents and brothers have always been there for me, even when things got tough. Their support means the world to me.

I would also like to thank my advisors, Dr. Dennis Lin and Dr. Lingzhou Xue, who played a huge role in my development at Penn State. Dr. Lin taught me many lessons, but the most critical one was the importance of thinking independently. Dr. Lin encouraged me to write more and more about my ideas, challenged me to explore new research avenues, and guided me through the daunting process of submitting publications. Dr. Xue offered invaluable insight into the world of academia. He always patiently listened to my ideas, and never hesitated to offer constructive suggestions. He gave me much-needed guidance and support when applying for academic positions.

I thank the members of my committee: Dr. Murali Haran, Dr. Yanyuan Ma, and Dr. Hui Yang. You have taken much time out of your busy schedules for me, and it is appreciated. The insight and perspective you have brought to this research has been very helpful.

I would like to thank Dr. Peter Winker for his guidance, patience, and flexibility. I know it was not always easy or convenient to schedule Zoom calls across multiple time zones. His technical expertise and insight was an invaluable asset to this research.

I would also like to thank Dr. Kevin Quinlan, and the other researchers at Lawrence Livermore National Laboratory that I had the pleasure of working with in my summer internship. My time at LLNL showed me the wide world of applications for statistics.

This dissertation research was partially supported by the U.S. National Science Foundation (NSF) via grant DMS-18102925. The findings and conclusions expressed in this work do not necessarily reflect the views of the funding agency.

Chapter 1
Introduction

In a mixture experiment, there are m components that are mixed together in a fixed total amount to produce a response y. It is typically assumed that the response only depends on the proportion of each ingredient that is included in the mixture. The objective of a mixture experiment is to find values of the mixture proportions that produce an optimal response. Typically, the objective of a mixture experiment is to find the values of the mixture proportions that maximize a response, minimize a response, or have the response match a pre-existing target value.

The focus of this dissertation is on Order-of-Addition (OofA) Mixture experiments and the various challenges they present. In these experiments, both the mixture proportions and the order in which the components are added to the mixture impact the response. If only the order of the m components is considered, the problem is still very daunting. There are $m!$ possible orderings of these components to consider, which scales rapidly for large m. For example, when $m = 10$, $m! \approx 3.6$ million possible orders. Bringing the mixture proportions into consideration also increases the scale of the experimental region. Suppose that a mixture design in m components has t mixtures of interest, where t depends on m. A simple approach to designing such an experiment would be to try all $t \times m!$ mixture-order combinations. Therefore, a major challenge posed by these experiments is the creation of efficient small-run designs. Another challenge is how one can handle the simultaneous modeling of mixture and order. Modeling mixture proportions and addition order separately is not desirable, as these effects may produce synergistic or antagonistic interactions on the response. Furthermore, even if suitable designs and models for the OofA experiment can be found, efficient methods for finding the optimal mixture and order are needed, especially in the case of mixture-order interactions. Finally, in many practical applications, separate constraints are often placed on mixture proportions and the order of addition, which can make the application of

traditional mixture or OofA designs more complex.

In general, there are many scientific applications where the order of the components produces an effect on the response. One major example is combinatorial drug therapy, where drugs can be given in particular combinations and ratios so that they have a synergistic effect at treating cancer (Al-Lazikani et al., 2012). Ding et al. (2015) provides a similar example in the field of combinatorial drug therapy, where both the ratio and order of three drugs were examined for treatment of oral cancer. Another example of order of addition in bioengineering is provided by Chandrasekaran et al. (2006), where the efficiency of synthesis of carbonate products depended on the order of addition of alcohols. An example of order of addition in the pharmaceutical industry is given by Rajaonarivony et al. (1993), where the size of nanoparticles formed was found to depend on the order of addition of poly-L-lysine and calcium to sodium-alginate solution.

There are many experiments where the response depends on both the mixture proportions and their addition order. Sljivic-Ivanovic et al. (2015) ran an experiment where they varied the order of $m = 3$ sorbents in a mixture, and examined how this impacted the removal of ions from aqueous solutions. Voelkel and Gallagher (2019) examined another example of OofA in a mixture-type experiment, where the addition order of binder resins and various additives were varied to study the impact of addition order on the viscosity of vehicle paint. In this study, any mixture components involved were held constant, so it is not possible to infer whether the mixture proportions had an effect on the response. While it might seem practical to hold the mixture proportions constant to study the order of addition, it should be noted that doing so makes it impossible to determine if there is interaction between the order of addition and the mixture proportions. It is therefore useful to construct designs where both the order of addition and the mixture proportions are varied, so that experimenters can determine whether the response changes for particular combinations of addition orders and mixture proportions.

The existing literature on mixture designs ignores the effects of the order of addition of the mixture components. The research presented here is novel, as it considers the scenario where the response depends on both the mixture proportions of components and their order of addition. The research contributions provided are organized into three parts.

First, full designs are proposed for the OofA Mixture experiment, and an algorithm is provided for their construction. Models based on these designs are introduced which include (1) typical mixture model parameters (called “pure mixture” terms), and (2)

pairwise order-of-addition effects of the mixture proportions. Additionally, models with interaction between the mixture proportions and the addition order are considered. An algorithm is proposed for finding the mixture proportions and addition order that produce an optimal response. Empirical simulations demonstrate that if there is interaction between the mixture proportions and their addition order, then the optimal mixture proportions identified by traditional models may be misleading. Finally, a Simulated Annealing (SA) algorithm is proposed for efficiently searching the experimental region for optimal mixture proportions and order, given a fitted OofA Mixture model.

Second, various means of finding optimal OofA Mixture designs are proposed. It is noted that the full OofA Mixture designs have a very large number of runs. This means that if the number of mixture components is very large, then running a full OofA Mixture design may be inefficient or simply too costly. It is desirable to reduce the number of runs in an OofA Mixture experiment, while also paying attention to optimality criteria, such as D- or I-optimality. Both the Federov exchange algorithm and the Threshold Acceptance (TA) algorithm are adapted to search for $D-$ and $I-$optimal subsets of n rows of full OofA Mixture design. To implement the TA algorithm, neighborhoods for points in an OofA Mixture design are proposed for the Simplex-Lattice design, and then generalized to arbitrary simplex designs. Examples comparing the performance of these two algorithms is provided. It is shown that highly efficient designs can be found at low cost, even in the presence of constraints on the mixture proportions.

Third, designs are found in the case where the set of possible orders of the mixture components is constrained by a network or graph. In many chemical experiments, it is common for certain reactants to only be mixed with "neighboring" reactants for safety or logistical reasons. In this scenario, the experimental region is the set of all Hamiltonian paths on an undirected graph of m vertices. Depth-First Search (DFS) is used to enumerate full designs of all possible permutations on a given graph. Sufficient conditions on the graph for estimability of pairwise ordering model parameters are derived. A fractional DFS design is proposed that has desirable relative $D-$ and $A-$ efficiency to the full design under several simulated graphical models. Applications of the proposed methodology are shown for the automotive and pharmaceutical industries.

This dissertation is organized as follows. First, a detailed overview of optimal design, OofA experiments, and Mixutre experiments is provided in Chapter 2. Then, in Chapter 3, full OofA Mixture designs and models are proposed, followed by a simulation study which investigates the behavior of optimal points when there are mixutre-order interactions. In Chapter 4, various approaches to constructing optimal OofA Mixture designs with

reduced run sizes are discussed. In Chapter 5, the scenario where the set of possible orders is constrained by a graph is explored. Finally, Chapter 6 provides a conclusion and a summary of future work.

Chapter 2
Literature Review

This chapter provides a comprehensive review of existing literature on relevant experimental designs. Important background on Order-of-Addition (OofA) experiments is reviewed in Section 2.1. This is followed by an overview of mixture experiments, with information provided on common mixture designs in Section 2.2. In both of these experimental environments, it is of interest to identify designs with a small number of runs with desirable properties. The traditional criteria for optimal experimental designs are reviewed in Section 2.3.

2.1 Order-of-Addition Experiments

In an Order-of-Addition (OofA) experiment, the response depends on the order in which m components are sequentially added to a system. As such, there are $m!$ possible treatments; i.e., there are $m!$ permutations of the values $(1, 2, \ldots, m)$. The number of possible orderings increases dramatically as the number of components increase. This presents a challenge when searching for an optimal order, as well as finding optimal fractions of the $m!$ possible orders to use in an experiment.

2.1.1 Pairwise Ordering (PWO) Model

Most of the existing work on the OofA problem is on the Pair-Wise Ordering (PWO) model, which was introduced by Van Nostrand (1995). It was officially called the Pair-Wise Ordering (PWO) model in Voelkel (2019). Notation from Lin and Peng (2019) will be used here. Suppose that there are m components $1, 2, \ldots, m$ and a permutation is represented by $\mathbf{a} = (a_1, \ldots, a_m)^T$. Let $\mathcal{P}$ be the set of all pairs (j, k) where $1 \leq j < k \leq m$.

Let jk denote the pair (j, k). The PWO factor for all $jk \in \mathcal{P}$ is defined as

$$z_{jk}(\mathbf{a}) = \begin{cases} 1 & \text{if } j \text{ precedes } k \text{ in } \mathbf{a} \\ -1 & \text{if } k \text{ precedes } j \text{ in } \mathbf{a} \end{cases} \tag{2.1}$$

An example of the PWO variable coding scheme is provided in Table 2.1 for $m = 3$. In this case, there are $3! = 6$ possible orderings of the components. These orderings are listed in the leftmost column. For each ordering $\mathbf{a}$, the PWO variables $z_{12}(\mathbf{a}), z_{13}(\mathbf{a})$, and $z_{23}(\mathbf{a})$ are listed. For example, if $\mathbf{a} = (3, 1, 2)$ then $z_{12}(\mathbf{a}) = 1, z_{13}(\mathbf{a}) = -1, z_{23}(\mathbf{a}) = -1$.

Table 2.1: OofA Design for $m = 3$, all possible orders

Order	$z_{12}(\mathbf{a})$	$z_{13}(\mathbf{a})$	$z_{23}(\mathbf{a})$
(1,2,3)	1	1	1
(1,3,2)	1	1	−1
(3,1,2)	1	−1	−1
(3,2,1)	−1	−1	−1
(2,3,1)	−1	−1	1
(2,1,3)	−1	1	1

Unlike a 2^m factorial design, not all possible combinations of the PWO factors are feasible. The PWO factors must obey the transitive property. For instance, if $z_{ij}(\mathbf{a}) = 1$ and $z_{jk}(\mathbf{a}) = 1$, then it must be true that $z_{ik}(\mathbf{a}) = 1$. Thus, certain combinations are impossible, such as $z_{12}(\mathbf{a}) = 1, z_{13}(\mathbf{a}) = -1, z_{23}(\mathbf{a}) = 1$. This makes traditional full or fractional factorial design approaches impractical, as the transitive property must be considered for any triplet of components j, k, ℓ where $j < k < \ell$.

Let $\tau(\mathbf{a})$ be the expected response given permutation $\mathbf{a}$. The PWO model is

$$\tau(\mathbf{a}) = \beta_0 + \sum_{jk \in \mathcal{S}} z_{jk}(\mathbf{a})\delta_{jk} \tag{2.2}$$

A parameter estimate $\hat{\delta}_{jk}$ indicates how the pairwise order of components j and k impacts the expected response. If certain parameters were identified as significant, then these parameters would provide clues to help determine the optimal order. In Lin and Peng (2019), topological sorting methods are discussed for finding the optimal order given the output of the PWO model. There are also several results concerning the optimality of PWO designs. Let the moment matrix M be defined as $M = X^T X/n$, where n is the number of runs in the PWO design (the unreplicated full design would have $n = m!$). Peng et al. (2019) showed that the full design with all $m!$ runs is optimal

for $D-, A-, E-$, and $M.S.-$ criteria, as well as for any criteria that is concave and signed permutation invariant.

2.1.2 Component-Position (CP) Model

There are some proposed alternatives to the popular PWO coding scheme. One of these is the Component-Position (CP) model, as described in Yang et al. (2021). This model relies on CP factors, which are defined as

$$z_c^{(j)}(\mathbf{a}) = \begin{cases} 1 & \text{if component } c \text{ is in position } j \text{ in } \mathbf{a} \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

An example of the CP coding scheme is provided in Table 2.2 for all 6 possible orders of $m = 3$ components. Table 2.2 illustrates that for a fixed component c, it must be true that $\sum_{j=1}^{m} z_c^{(j)} = 1$. This is because each component must occupy some position in a permutation, and a component cannot appear more than once. For example, if $\mathbf{a} = (3, 1, 2)$, then $z_1^{(2)} = 1$ and $z_1^{(1)} = z_1^{(3)} = 0$. Similarly, for fixed j, we must have that $\sum_{c=1}^{m} z_c^{(j)} = 1$, position j can only be filled by one component.

Table 2.2: CP Design for $m = 3$, all possible orders

Order	$z_1^{(1)}$	$z_1^{(2)}$	$z_1^{(3)}$	$z_2^{(1)}$	$z_2^{(2)}$	$z_2^{(1)}$	$z_3^{(1)}$	$z_3^{(2)}$	$z_3^{(3)}$
(1,2,3)	1	0	0	0	1	0	0	0	1
(1,3,2)	1	0	0	0	0	1	0	1	0
(3,1,2)	0	1	0	0	0	1	1	0	0
(3,2,1)	0	0	1	0	1	0	1	0	0
(2,3,1)	0	0	1	1	0	0	0	1	0
(2,1,3)	0	1	0	1	0	0	0	1	0

Using the CP factors, a linear model for the effect of addition order on the response can be formed.

$$\tau(\mathbf{a}) = \mu_0 + \sum_{c=1}^{m} \sum_{j=1}^{m} \delta_c^{(j)} x_c^{(j)}(\mathbf{a}) \tag{2.4}$$

In Model (2.4), $\tau(\mathbf{a})$ is the expected response for a permutation $\mathbf{a}$, $x_c^{(j)}(\mathbf{a})$ is the CP variable for component c and position j of permutation $\mathbf{a}$. We can interpret $\delta_c^{(j)}$ as the effect of placing component c in position j on the expected response. Due to the constraints $\sum_{j=1}^{m} z_c^{(j)} = 1$, $\sum_{c=1}^{m} z_c^{(j)} = 1$, the model matrix resulting from fitting

(2.4) directly will not have full rank. Therefore, constraints must be placed on the model parameters $\delta_c^{(j)}$ to ensure identifiability. One option is zero-sum constraints, where $\sum_{j=1}^m \delta_c^{(j)} = \sum_{c=1}^m \delta_c^{(j)} = 0$. Another are baseline constraints, where $c = 1$ and $j = m$ are taken as baselines, and $\delta_1^{(j)} = 0$ for all $j = 1, 2, \ldots, m$ and $\delta_c^{(m)} = 0$ for all $c = 1, 2, \ldots, m$.

Although the bulk of existing results regarding optimal OofA designs are established under the PWO model, several exist for the CP model. For instance, Yang et al. (2021) point out that the $\phi-$optimality of the full design with all $m!$ orders also generalizes to the CP model. The CP model can also be used to make effective predictions of the response given the order, so it is useful in cases where all $m!$ orders can be enumerated for small m.

2.1.3 Optimal Fractional OofA Designs

The full OofA designs in Peng et al. (2019) are $\phi-$optimal, but it is too expensive to run all $m!$ trials for large m. Peng et al. (2019) showed a systematic way to construct some optimal fractional designs. In particular, a fractional PWO design is optimal iff it has the same moment matrix as the full design. These fractional designs are of size $m!/s!, s = \lfloor m/2 \rfloor$. The following example illustrated how this design is constructed when $m = 4$.

To do this, denote $C_1 = \{1, 2\}, C_2 = \{1, 3\}, C_3 = \{1, 4\}$ as the sets of all groups of s components in increasing order that begin with component 1. Using these, one constructs matrices

$$B_1 = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}, \quad \bar{B}_1 = \begin{pmatrix} 3 & 4 \\ 4 & 3 \end{pmatrix}, \quad B_2 = \begin{pmatrix} 1 & 3 \\ 3 & 1 \end{pmatrix}, \quad \bar{B}_2 = \begin{pmatrix} 2 & 4 \\ 4 & 2 \end{pmatrix}$$

$$B_3 = \begin{pmatrix} 1 & 4 \\ 4 & 1 \end{pmatrix}, \quad \bar{B}_3 = \begin{pmatrix} 2 & 3 \\ 3 & 2 \end{pmatrix}$$

Each B_u is constructed by taking all permutations of C_u, $u = 1, 2, 3$ in lexicographical order. Each $\bar{B}_u$ contains all permutations of $\bar{C}_u = \{1, \ldots, m\} \setminus C_u$ for $u = 1, 2, 3$. Once this is done, then the fractional OofA design is $[D_1^T, D_2^T, D_3^T]^T$ where

$$D_u = \begin{pmatrix} B_u & \bar{B}_u \\ \sim \bar{B}_u & B_u \end{pmatrix}, \quad u = 1, 2, 3$$

where $\sim$ denotes column reversal. Intuitively, this construction method is optimal

because this blocking scheme ensures that each pair of components will appear forward and backward the same number of times, while also scaling down the number of runs required. It is not actually necessary to choose all groups of s components as shown in the above example. In particular, Chen et al. 2020 generalized construction of these designs to any set of blocks $C_1, \ldots, C_b$ that come from a balanced incomplete block design in m symbols.

2.1.4 TA Algorithm for OofA Designs

Another method for finding optimal subsets of the full OofA design is to use a search algorithm. One good example of this is provided by Winker et al. (2020), where the Threshold Accepting (TA) was used to search for a subset of n runs with high $D-$efficiency. In this work, it was of interest to find efficient minimal-point, double-point, and triple-point designs. If a model has p parameters, then these designs are of size $n = 1 + p, n = 1 + 2p, n = 1 + 3p$, respectively. In a minimal-point design, the additional run is typically used to estimate the error variance. It should be noted that this algorithm is not limited to these specific values of n, provided that $n > 1 + p$. For the PWO model, $p = \binom{m}{2}$. Pseudocode for this algorithm is provided in Algorithm 1.

Algorithm 1: Pseudo code for generic TA algorithm for finding D-optimal design.

Input: A sequence $\{t_r : r = 1, 2, \ldots\}$ of real numbers such that $t_r \to 0$ as $r \to \infty$.

1. Initialize a starting design D_0. Set $r = 1$.

while *a convergence condition is not met* **do**

2. Find $D_1 \in \mathcal{N}(D_0)$.
3. **if** $de(D_1) > de(D_0) - t_r$ **then**
   $D_0 = D_1$

end

4. r = r+1;

end

5. Return D_0.

As inputs, the Algorithm 1 takes a sequence $\{t_r : r = 1, 2, \ldots\}$ that tends to zero as r approaches infinity. This sequence is referred to as the threshold sequence. When the algorithm begins, a starting design D_0 is initialized on Line 1. This design is of the target size n, and it is typically chosen through randomization. If n is small, it may be necessary to check that this matrix leads to a nonsingular model matrix; if it does, resampling can be done until a suitable D_0 matrix is found. Then, during each iteration

of the TA algorithm, a design D_1 is randomly selected from a "neighborhood" of designs around D_0, denoted as $\mathcal{N}(D_0)$ in Line 2. The design D_1 should be similar to D_0 apart from a small, random perturbation in one or two of its rows. In Line 3, the $D-$efficiency of D_1, denoted as $de(D_1)$, is compared to that of D_0. If D_1 is more efficient than D_0 within the threshold of t_r, then D_0 is updated to D_1. This process is repeated until a convergence criteria is met; e.g., until the marginal improvement in $D-$optimality is below some threshold.

The use of thresholds in the TA algorithm helps prevent it from becoming trapped at local optima; in this case, maxima. At the beginning of the algorithm, the thresholds t_r will be large. This means that when the comparison in Line 3 is made, some sub-optimal updates will be made. This will prevent the algorithm from prematurely converging to a local maximum. If the values of t_r are properly tuned, then these changes will not be so detrimental as to lead the algorithm away from the global optimum. As the algorithm continues, t_r will decrease, which results in fewer sub-optimal updates to the design. Eventually, when r is very large, t_r will be close enough to zero that the algorithm will only choose updates that strictly improve the efficiency of the design.

To implement a TA algorithm, a definition of the neighborhood $\mathcal{N}$ and a method for finding the thresholds $\{t_r : r = 1, 2, \ldots, \}$ are both needed. Winker et al. (2020) define $\mathcal{N}(D_0)$ as the set of designs that are identical to D_0 apart from two rows, where the permutations in each of the two rows have had one pair of adjacent components swapped. In general, the number of rows that can be modified is flexible, but Winker et al. (2020) chose to use two. An empirical algorithm is used to find the threshold sequence; more details on this are explored in Chapter 4, where the TA algorithm is visited again in Chapter 4 as a means for finding more efficient OofA Mixture designs.

2.2 Mixture Designs

Let $x_1, \ldots, x_m$ represent the proportions of the m components in a mixture experiment, with $0 \leq x_i \leq 1$ for $i = 1, \ldots, m$ and $\sum_i x_i = 1$. The objective of a mixture experiment is to find values of the mixture component proportions $x_1, \ldots, x_m$ that optimize (maximize or minimize) the response y. Alternatively, the objective can be to choose $x_1, \ldots, x_m$ to match a target response T. Note that the mixture components take values in the $(m-1)$ dimensional simplex $\mathcal{S} = \{(x_1, \ldots, x_m) \in [0, 1] | \sum_i x_i = 1\}$. We first review three classical mixture designs: the simplex-lattice design, and the simplex-centroid design, and the extreme-vertices design (Cornell, 1990). For any of these designs, it is often

sufficient to use a second-order polynomial to model the response surface:

$$\eta = \sum_{i=1}^{m} \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j \tag{2.5}$$

where η represents the response surface and the x_i is the value taken by the i^{th} mixture component. The model has no intercept term due to the constraint that $\sum_i x_i = 1$. Cornell (1990) also shows that this constraint may also be used to eliminate pure quadratic terms, i.e. $\beta_{ii} x_i^2$ from the model. It is rare to see an application use a degree higher than second-order.

2.2.1 Simplex-Lattice Design

The $\{m, \ell\}$ simplex lattice design, called the $\{q, m\}$ simplex lattice design by Cornell (1990), is a design for m components and a polynomial model of degree l for the response surface. It uses design points of the form $x_i = 0, \frac{1}{\ell}, \frac{1}{\ell}, \frac{2}{\ell}, \ldots 1$. The design consists of all possible combinations of the of these x_i that produce points in the simplex $\mathcal{S}$.

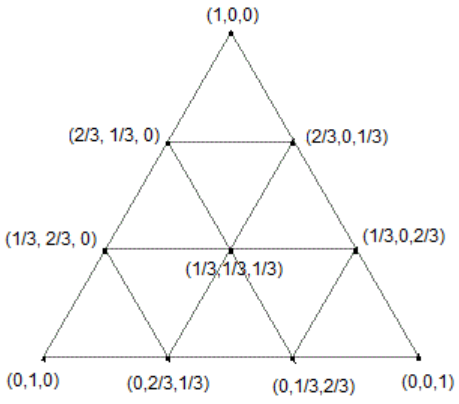


Figure 2.1: A $\{3, 3\}$ Simplex Lattice Design.

Figure 2.1 shows an example of a simplex lattice design for $m = 3, \ell = 3$. Each point in the lattice satisfies $\sum_{i=1}^{m} x_i = 1$. Also, each point in the lattice is equally spaced. In general, the space between any pair of points in a simplex lattice design is $\frac{\sqrt{2}}{\ell}$. This design (as well as the simplex centroid design below) often have a majority of points on the boundary of the simplex. These boundary points represent mixtures that do not use all of their components. This makes the simplex-lattice design a classical choice for a mixture experiment where such points are realistic.

2.2.2 Simplex-Centroid Design

The simplex-centroid design, also covered in Cornell (1990), has a total of $2^m - 1$ design points. The design points are generated as follows. Start by taking all m permutations of the point $(1, 0, \ldots, 0)$. Then, take all $\binom{m}{2}$ permutations of $(1/2, 1/2, 0, \ldots, 0)$, all $\binom{m}{3}$ permutations of $(1/3, 1/3, 1/3, 0, \ldots, 0)$, and so on until one reaches the point $(1/m, \ldots, 1/m)$, which is the centroid. Unlike the simplex lattice design, the maximal degree of the response surface does not need to be specified to create a simplex centroid design; only the number of components m needs to be provided.

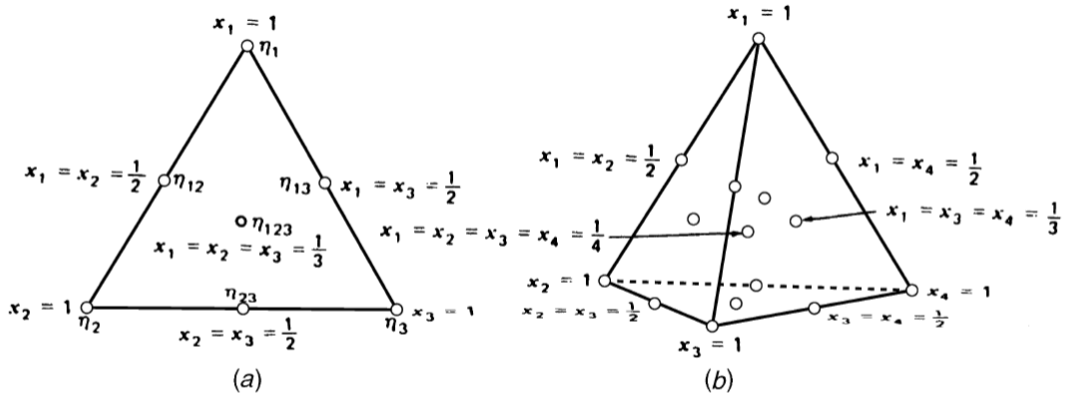


Figure 2.2: Simplex-Centroid Designs from Cornell (1990). Left: $m = 3$, Right: $m = 4$.

Figure 2.2 shows Simplex-Centroid designs for $m = 3$ (left panel) and $m = 4$ (right panel). In Figure 2.2, the value of any component at a point that is not listed is taken to be zero. For example, at point η_2 in the left panel, $(x_1, x_2, x_3) = (0, 1, 0)$. In general, these designs only have one point strictly in the interior of the simplex, which is the centroid. All other points are on the vertices, edges, or faces of the simplex. For example, When $m = 4$, only the centroid $(x_1, x_2, x_3, x_4) = (1/4, 1/4, 1/4, 1/4)$ is in the interior, while the remaining points have at least one $x_i = 0$ for some $i = 1, 2, 3, 4$. It is common for the Simplex-Centroid design to be augmented with more points on the interior if more information about the response surface is required in this region.

2.2.3 Extreme Vertices Design

Sometimes, there are constraints placed on the mixture proportions. For example, practical concerns might make it so the first component must be between 10% and 80%. In this case, we assume that there are single-component constraints $0 \leq L_i \leq X_i \leq U_i \leq 1$, for $i = 1, 2, \ldots, m$. In this case, the region of interest is not the entire simplex, but a

polyhedron that lies within the simplex. This design uses the vertices of the polyhedron, in addition to points positioned on the center of any faces of the region and an overall centroid. These points can be systematically identified by several procedures, such as the XVERT algorithm provided by Snee and Marquardt (1974).

In order to fit a mixture design to a polyhedron defined by single-component constraints, it is necessary to first check that the constraints are consistent, i.e., satisfiable. According to Piepel (1983), a set of constraints $0 \leq L_i \leq x_i \leq U_i \leq 1$ is consistent if it is possible for each proportion x_i to attain its lower bound L_i and its upper bound U_i while still satisfying the constraints. For example, the constraints

$$0.20 \leq x_1 \leq 0.40$$
$$0.20 \leq x_2 \leq 0.60$$
$$0.18 \leq x_3 \leq 0.70$$

are inconsistent because it is not possible to attain $x_3 = U_3 = 0.70$, as that would imply that $x_1 + x_2 = 0.30$; therefore, it must be true that either $x_1 < 0.20$ or $x_2 < 0.20$, which violates the lower bound of the first two constraints. Crosier (1984) provides a method for using $U-$pseudocomponents for fixing inconsistent constraints with their implied upper bounds $U_i^* = 1 - \sum_{j \neq i} L_j$. For example, if U_3 is replaced by $U_3^* = 1 - L_1 - L_2 = 0.6$, then the resulting set of constraints will be consistent.

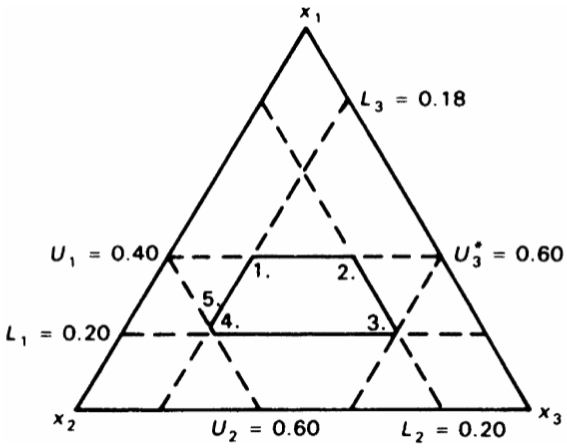


Figure 2.3: Polyhedron Defined by $0.2 \leq x_1 \leq 0.4, 0.2 \leq x_2 \leq 0.6, 0.18 \leq x_3 \leq 0.60$, from Cornell (1990)

An example of the region defined by these consistent single-component constraints is shown in Figure 2.3, which is taken from Cornell (1990). The constrained region is no

longer a simplex, but a polyhedron. Hence, a standard simplex design is no longer feasible in this case. Throughout this dissertation, we assume that any constraints placed on the mixture proportions are consistent. If they are not, then the methods in Piepel (1983) and Crosier (1986) can be applied to adjust the constraints so that they are consistent.

2.3 Optimal Design Criteria

In both Mixture and Order-of-Addition experiments, we will encounter linear regression models of the form

$$y = X\theta + \epsilon$$

where $X \in \mathbb{R}^{n\times p}$ is called a model matrix, $\theta \in \mathbb{R}^{p\times 1}$ is a vector of parameters, $y \in \mathbb{R}^{n\times 1}$ is a vector of responses, and $\epsilon \in \mathbb{R}^{n\times 1}$ is a vector of errors. We assume that $E[\epsilon] = 0, \text{Var}(\epsilon) = \sigma^2 I_p$; i.e., that the errors have mean zero and are homoscedastic. In this case, the least-squares estimator for θ and its corresponding variance are

$$\hat{\theta} = (X^T X)^{-1} X^T y$$
$$\text{Var}(\hat{\theta}) = \sigma^2 (X^T X)^{-1}$$

It is desirable to "minimize" the variance of the least squares estimator $\hat{\theta}$ in some sense. The true variance σ^2 is unknown in most applications. From a design perspective, we only have control of the model matrix X, which is formed by taking runs from a design matrix D and applying a model expansion. Therefore, we must choose the matrix D, and by extension, X, in a way that minimizes this variance.

2.3.1 D-Optimality

The $D-$optimality criterion was first introduced by Wald (1943) as a means of maximizing power in an $F-$test ANOVA setting. This idea was extended to general regression models by Kiefer and Wolfowitz (1959). The $D-$optimality criterion is given by

$$X^* = \arg\min_{X\in\mathcal{M}} |(X^T X)^{-1}| = \arg\max_{X\in\mathcal{M}} |X^T X|$$

where $\mathcal{M}$ is the space of all possible model matrices. This criterion seeks to minimize the determinant of the information matrix $(X^T X)^{-1}$. As argued by Kiefer and Wolfowitz

(1959), there are several benefits to using the $D-$ optimality criterion. First, minimizing the determinant of the information matrix is equivalent to minimizing the volume of a confidence ellipsoid about the vector $\hat{\theta}$. The $D-$optimality criteion is also invariant under scalar multiplication and other transformations, such as multiplication by signed permutation matrices. Finally, it was subsequently shown that $D-$optimality is actually equivalent to $G-$optimality, which was one of the very first optimality criteria introduced (Smith, 1918). This criteria can be written as

$$X^* = \arg\min_{X \in \mathcal{M}} \max_{x \in \mathcal{X}} \text{Var}(\hat{y}(x))$$

where $\mathcal{X}$ is the space of all possible values of a predictor x. In other words, the $G-$optimality criterion seeks to minimize the maximum variance of a predicted response. The equivalence of $G-$ and $D-$optimality is commonly referred to as the equivalence theorem.

2.3.2 A-Optimality

Another criteria of interest $A-$optimality was introduced by Chernoff (1953), who was interested in finding locally optimal designs in an experimental setting. This criterion is given by

$$X^* = \arg\min_{X \in \mathcal{M}} \text{trace}((X^T X)^{-1})$$

This can be interpreted as minimizing the sum of the variances across all predictors in $\hat{\theta}$. This criterion is appealing because of the simple interpretation, and the idea that minimizing the sum of the variances also minimizes their average for fixed p. While conceptually simpler than $D-$optimality, the $A-$optimality criterion is often more computationally burdensome, since it is required that the inverse of $X^T X$ is computed. $D-$optimality can avoid this by maximizing the determinant of the matrix before inversion.

2.3.3 I-Optimality

A common goal in analysis is prediction of the response at many points in the experimental region $\mathcal{X}$. The variance of the predicted response at a point $\mathbf{x} \in \mathcal{X}$ under a model f is

given by

$$\text{Var}(\hat{y}(\mathbf{x})) = f(\mathbf{x})^T (X^T X)^{-1} f(\mathbf{x})$$

where $f(\mathbf{x}) \in \mathbb{R}^{p \times 1}$ is the model expansion of an input $\mathbf{x} \in \mathcal{X}$. In many experiments, it is desirable to minimize this prediction variance over a well-defined region. In this case, it is desirable to integrate the prediction variance over the region $\mathcal{X}$ (Box and Draper, 1963; Hardin and Sloane, 1993). This is particularly the case in mixture experiments, where the input space is constrained to an $(m-1)$dimensional simplex (see Section 2.2). The $I-$optimality criterion is defined as

$$X^* = \underset{X \in \mathcal{M}}{\arg\min} \frac{\int_{\mathcal{X}} f(\mathbf{x})^T (X^T X)^{-1} f(\mathbf{x}) d\mathbf{x}}{\int_{\mathcal{X}} d\mathbf{x}} \tag{2.6}$$

The numerator of (2.6) is the integrated prediction variance over the experimental region $\mathcal{X}$. The denominator is the total volume of the region. These designs are desirable for applications where a response surface needs to be estimated over a region of interest. The numerator in (2.6) is often rewritten using the fact that the trace is invariant to cyclic matrix permutations:

$$\begin{aligned} X^* &= \underset{X \in \mathcal{M}}{\arg\min} \frac{\text{trace}((X^T X)^{-1} B)}{\int_{\mathcal{X}} d\mathbf{x}} \\ B &= \int_{\mathcal{X}} f(\mathbf{x}) f(\mathbf{x})^T d\mathbf{x} \end{aligned}$$

The integration required to find the matrix B has a simple analytic closed form if $\mathcal{X}$ is a regular $(m-1)$ dimensional simplex, which can be derived using the degenerate Dirichlet distribution (DeGroot, 1970; Goos et al., 2016). There is no general closed form of B for an arbitrary space $\mathcal{X}$, which can make finding $I-$optimal designs challenging.

Chapter 3
Order-of-Addition Mixture Experiments

3.1 Introduction

The OofA mixture experiment is a mixture experiment where the researchers are also concerned that the order in which components are added into the mixture has an effect on the response. In such studies, the response surface would be a function of both the mixture proportions $x_1, \ldots, x_m$ and their order of addition. In this experiment, there are three goals of interest: (1) Determine whether the order-of-addition of the components has a statistically significant effect on the response; (2) Determine whether the mixture components have a statistically significant effect on the response; (3) Find the settings (i.e., mixture proportions and ordering) that gives an optimal response; i.e., a response that is either maximized, minimized, or matches a target value T. In this chapter, we focus on constructing designs and models that allow researchers to simultaneously achieve both of the above goals.

In this chapter, we first formulate the OofA Mixture problem. We show how to construct designs for this problem in a systematic fashion. Theoretical properties of this design are explored. We also show how to model the response surface, where the response depends on both the mixture proportions and their order of addition. A detailed simulation study demonstrates that, if there are significant mixture-order interactions, the contours of the response surface can significantly change. In this case, traditional mixture models which ignore the order of addition yield misleading results.

3.2 Problem Formulation

Suppose that there are m components that will be added into a mixture with a fixed total amount to produce a continuous response y. Let $x_1, \ldots, x_m$ be the proportions of each component that are included in the mixture. Let $\mathcal{A}$ be the set of all permutations of $(1, 2, \ldots, m)$ and $\mathcal{S} = \{(x_1, \ldots, x_m) \in [0, 1] | \sum_i x_i = 1\}$ or a sub-region of this simplex determined by single-component constraints. In the OofA Mixture experiment, it is assumed that the response depends on both the mixture proportions and their order of addition. This can be expressed as

$$y = f(\mathbf{x}, \mathbf{a}) + \epsilon$$

where $\epsilon \sim N(0, \sigma^2)$, $\mathbf{x} \in \mathcal{S}$, and $\mathbf{a} \in \mathcal{A}$. There are two goals to keep in mind when designing an OofA mixture experiment. First, the experimental design should ensure that effects only due to mixture components (pure mixture effects) have low correlation with effects that involve the order of addition. It is desirable for these effects to be orthogonal, but this may not always be the case. In Appendix 3.A, we prove that these effects are indeed orthogonal if the full design from Section 3.4.2 is used. Second, the design and accompanying model should allow us to find the optimal mixture proportions and ordering. In the case of finding a maximum (or a minimum), the optimal proportions and ordering are

$$(\mathbf{x}^*, \mathbf{a}^*) = \arg\max_{\mathbf{x},\mathbf{a}} f(\mathbf{x}, \mathbf{a})$$
$$\text{subject to} \quad \mathbf{a} \in \mathcal{A} \quad \text{and} \quad \mathbf{x} \in \mathcal{S}$$

In the case of matching a target value T, the optimal proportions and ordering are

$$(\mathbf{x}^*, \mathbf{a}^*) = \arg\min_{\mathbf{x},\mathbf{a}} (f(\mathbf{x}, \mathbf{a}) - T)^2$$
$$\text{subject to} \quad \mathbf{a} \in \mathcal{A} \quad \text{and} \quad \mathbf{x} \in \mathcal{S}$$

The above are constrained optimization problems, since the optimal $\mathbf{x}^*$ must lie in the simplex, and the optimal ordering must be a valid permutation of $(1, 2, \ldots, m)$. For the purpose of this initial research we focus on maximizing or minimizing the response, as the problem of matching a target T is essentially minimizing a function of the response.

3.3 Construction of the Full Design Matrix

In this section, novel OofA Mixture designs are constructed based on a simplex designs. Let $\mathcal{P} = \{jk \mid j, k \in (1, \ldots, m),\ j < k\}$ and $\mathcal{S}$ be an $(m-1)$ dimensional simplex. Then for $\mathbf{x} \in \mathcal{S}$, $jk \in \mathcal{P}$, and a permutation $\mathbf{a}$ of $(1, \ldots, m)$, we define the modified PWO variables

$$z_{jk}(\mathbf{x}, \mathbf{a}) = \begin{cases} 1 & x_j, x_k \neq 0 \text{ and } j \text{ is before } k \text{ in } \mathbf{a} \\ 0 & x_j = 0 \text{ or } x_k = 0 \\ -1 & x_j, x_k \neq 0 \text{ and } j \text{ is after } k \text{ in } \mathbf{a} \end{cases}$$

If all of the mixture components are used, i.e., all $x_i > 0$ for $i = 1, 2, \ldots, m$, then the modified PWO components reduce to the original ones. The purpose of modifying the original PWO variables is to deal with edge cases where some of the $x_i = 0$. These modified PWO variables describe the ordering of the nonzero mixture components. Simplex designs, like the SLD or Simplex Centroid, have many vertex or edge points in the simplex where some mixture proportions are not used. This modification to the PWO variables ensures that orderings are not assigned to components that are not included in a row of the design matrix. For example, if $x = (0.5, 0.5, 0)$ and $a = (1, 2, 3)$, then $z_{12}(x, a) = 1$, but $z_{13}(x, a) = z_{23}(x, a) = 0$. Using these modified PWO variables, it is possible to construct a design matrix using Algorithm 2.

Algorithm 2: Generate Full OofA Mixture Design Matrix

Create a simplex design for m components, $SD(m)$.
Initialize a design matrix D.
for *each row* $\boldsymbol{x}$ *of* $SD(m)$ **do**
  Let k be the number of nonzero components of $\mathbf{x}$.
  Replicate $\mathbf{x}$ $k!$ times (including the original row).
  Associate each replicate with a unique ordering $\mathbf{a}$ of the k nonzero components of x.
  For each replicate, represent its ordering $\mathbf{a}$ using a row vector $\mathbf{z}$ of $\binom{m}{2}$ modified PWO variables $z_{jk}(\mathbf{x}, \mathbf{a})$ for each $jk \in \mathcal{P}$.
end
Stack the rows $\mathbf{x}$ (and their replicates) into a matrix X.
Stack the rows $\mathbf{z}$ into a matrix Z.
return $D = (X, Z)$

Algorithm 2 uses an existing mixture design $SD(m)$ as an input. Each row of $SD(m)$

is replicated once for every possible ordering of the nonzero components. Each replicate is then assigned an addition order in terms of the modified PWO variables. The designs generated by Algorithm 2 are called "full" designs because, for each row of $SD(m)$, they include one run for every possible ordering of the nonzero mixture components. For example, an OofA Mixture design constructed using a Simplex Lattice Design for $m = 3, \ell = 3$ is shown in Table 3.1.

Table 3.1: OofA Simplex Lattice Design, $m = 3, \ell = 3$

x_1	x_2	x_3	z_{12}	z_{13}	z_{23}
1.00	0.00	0.00	0.00	0.00	0.00
0.00	1.00	0.00	0.00	0.00	0.00
0.00	0.00	1.00	0.00	0.00	0.00
0.33	0.67	0.00	1.00	0.00	0.00
0.33	0.67	0.00	−1.00	0.00	0.00
0.67	0.33	0.00	1.00	0.00	0.00
0.67	0.33	0.00	−1.00	0.00	0.00
0.33	0.00	0.67	0.00	1.00	0.00
0.33	0.00	0.67	0.00	−1.00	0.00
0.67	0.00	0.33	0.00	1.00	0.00
0.67	0.00	0.33	0.00	−1.00	0.00
0.00	0.33	0.67	0.00	0.00	1.00
0.00	0.33	0.67	0.00	0.00	−1.00
0.00	0.67	0.33	0.00	0.00	1.00
0.00	0.67	0.33	0.00	0.00	−1.00
0.33	0.33	0.33	1.00	1.00	1.00
0.33	0.33	0.33	1.00	1.00	−1.00
0.33	0.33	0.33	1.00	−1.00	−1.00
0.33	0.33	0.33	−1.00	−1.00	−1.00
0.33	0.33	0.33	−1.00	−1.00	1.00
0.33	0.33	0.33	−1.00	1.00	1.00

In general, the number of rows N in D for the Full OofA Simplex Centroid Design is

$$N = m + \binom{m}{2}2! + \binom{m}{3}3! + \cdots + \binom{m}{m-1}(m-1)! + m!.$$

So N grows very quickly with m when D is constructed from the Simplex Centroid Design. It should be noted that the previous algorithm is a general framework; the rows x do not have to be drawn from the Simplex Centroid Design. They can be drawn from any unreplicated mixture design (e.g. Simplex Lattice). If there are single-component constraints, the rows may be drawn from an extreme vertices design. If the constrained region is in the interior of the simplex, the modified PWO variables will reduce to the PWO variables from Peng et al. (2019).

As m increases, the run size of the full design also grows quickly for the Simplex Lattice design, and even more so for the extreme vertices design. In the case of strict financial limitations on the number of runs, the full OofA Mixture design can be used as a list of candidate points for a smaller design of size $n < N$. Modern statistical packages

can be used to choose n of the N rows that maximize an optimality criterion, such as such as D-optimality, which maximizes the determinant of the information matrix. For instance, this can be done using the R package AlgDesign by Wheeler (2019). In cases where the number of mixture components is large, enumerating a set of candidate points may be burdensome, as such a set would be quite large. This subset of n rows is not guaranteed to be the optimal design, but these designs generally have desirable optimality criteria. In Chapter 4, methods for finding more optimal subsets of size n are explored in detail.

3.4 Models for OofA Mixture

In this section, we construct models for the pairwise ordering and mixture component effects. First, an additive model is considered, i.e., a model with no interactions between the mixture terms and the ordering:

$$y = X\beta + Z\delta + \epsilon \tag{3.1}$$

where $\epsilon \sim N(0, \sigma^2 I)$, β contains the coefficients for the mixture model, δ contains the coefficients for the PWO model, and X, Z are as defined in Algorithm 2. Model (3.1) is useful for identifying significant mixture or order main effects. However, if there is concern that the mixture effects depend on the addition order (or vice-versa), then Model (3.1) should be compared with a model that includes mixture-order interactions. Consider the following more general model:

$$y(\mathbf{x}, \mathbf{z}) = \eta(\mathbf{x}) + g(\mathbf{x}, \mathbf{z}) + \epsilon \tag{3.2}$$

where $\eta(\mathbf{x})$ models the response surface in terms of the mixture proportions, and $g(\mathbf{x}, \mathbf{z})$ is a function of both the mixture proportions and their order. Technically, Model (3.2) is a generalization of Model (3.1), i.e. take $\eta(\mathbf{x}) = X\beta$ and $g(\mathbf{x}, \mathbf{z}) = Z\delta$. However, with Model (3.2), interaction terms may now be included. For instance, consider:

$$y(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{m} \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j + \sum_{k<l} \delta_{kl} z_{kl} + \sum_{i} \sum_{k<l} \gamma_{kl}^{i} x_i z_{kl} + \epsilon \tag{3.3}$$

where γ_{kl}^{i} represents the effect of the interaction between mixture proportion x_i and the

order of components x_k, x_l. Model (3.3) is an instance of Model (3.2), where

$$\eta(\mathbf{x}) = \sum_{i=1}^{m} \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j \,, \quad g(\mathbf{x}, \mathbf{z}) = \sum_{k<l} \delta_{kl} z_{kl} + \sum_{i} \sum_{k<l} \gamma_{kl}^{i} x_i z_{kl}$$

Model (3.3) only includes "first order" interactions; i.e., interactions between single mixture proportions and PWO variables. It can be modified to include higher-order interactions, though this will clearly increase the number of parameters to estimate. In general, Model (3.3) will include $m + 2\binom{m}{2} + m\binom{m}{2}$ parameters. Another major flaw with Model (3.3) is that its model matrix will not have full rank. To see this, note that $z_{kl} = (x_1 + \cdots + x_m) z_{kl} = x_1 z_{kl} + \cdots + x_m z_{kl}$. So, each of the "main effects" z_{kl} can be written as a linear combination of the terms $x_1 z_{kl}, \ldots x_m z_{kl}$, so the columns of the resulting model matrix will be linearly dependent. This problem can be remedied in one of two ways. The first is by following methodology from Cornell (1990) and fitting a modification of model (3.3):

$$\eta(\mathbf{x}) = \sum_{i=1}^{m} \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j \;,\; g(\mathbf{x}, \mathbf{z}) = \sum_{i} \sum_{k<l} \gamma_{kl}^{i} x_i z_{kl} \tag{3.4}$$

Model (3.4) does not explicitly model the main effect of the order components; it models the effect of the order components only through their interactions with the mixture proportions. This is very similar to the form of the mixture models with process variables as shown in Chapter 7 of Cornell (1990). This "reduced" model only has $m + \binom{m}{2} + m\binom{m}{2}$ parameters. The second way to remedy this problem is to place restrictions on certain model parameters. When $m \geq 3$, assume that $\gamma_{kl}^{i} = 0$ if $i \neq k$ and $i \neq l$. In this case, the model becomes

$$\eta(\mathbf{x}) = \sum_{i=1}^{m} \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j \,, \quad g(\mathbf{x}, \mathbf{z}) = \sum_{k<l} \delta_{kl} z_{kl} + \sum_{i} \sum_{k<l, i=k, l} \gamma_{kl}^{i} x_i z_{kl} \tag{3.5}$$

Model (3.5) includes both the main PWO effects and some of their interactions with the mixture proportions. Specifically, the assumption $\gamma_{kl}^{i} = 0$ means that a mixture proportion only interacts with the pairwise orderings that it takes part in. For example, under Model (3.5), if $m = 3$ then mixture component 1 is may interact with z_{12} and z_{13}, but not z_{23}.

With an estimable model of the general form (3.2), the optimal proportions and order $\mathbf{x}^*, \mathbf{a}^*$ and response y^* can be found. This is done by finding the mixture proportions that produce a minimum (or maximum) response for each possible order, and then finding the

overall minimum response (or maximum) across the orders. Obviously, this is not ideal for larger m; rather, this algorithm simply outlines a brute force approach for finding an optimal value. Note that if the objective of optimization is to match a target T, then it is sufficient to minimize $(\hat{f}(\mathbf{x}, \mathbf{a}) - T)^2$ instead of $\hat{f}(\mathbf{x}, \mathbf{a})$.

3.4.1 Identifiability Constraints

If Model (3.4) or (3.5) is applied to an OofA Mixture design that is generated from a Simplex-Centroid design, then the resulting model matrix will also not be full rank. In the OofA Simplex-Centroid design, if x_j and x_k are mixture proportions for components j and k and z_{jk} is the indicator for the ordering of x_j and x_k, then notice that $x_j z_{jk} = x_k z_{jk}$. This is true because each row in the Simplex-Centroid design that satisfies $x_j \neq 0, x_k \neq 0$ must have $x_j = x_k$; otherwise, $z_{jk} = 0$, which preserves the equality. Since $x_j z_{jk} = x_k z_{jk}$, then some of the parameters in Model (3.4) or (3.5) will not be identifiable. Thus, for identifiability, it is required that for a simplex design SD, the set of design points $\{x \in SD \mid \ x_j \neq 0, x_k \neq 0, x_j \neq x_k, jk \in \mathcal{P}\}$ is non empty. This requirement is satisfied by several Simplex-Lattice Designs (e.g. {3,3} Simplex Lattice).

3.5 Theoretical Support

Here, theoretical results are shown for the models and designs presented in the previous sections. Proofs of these results may be found in Appendix 3.A at the end of this chapter.

Lemma 1 (Orthogonality of Mixture and PWO)**.** *Suppose that a design D is constructed using Algorithm 2. Then $D = (X, Z)$ such that X is a matrix with columns corresponding to the m mixture components, Z is a matrix with columns corresponding to the $\binom{m}{2}$ PWO variables, and $X^T Z = \mathbf{0}$.*

Lemma 1 follows from the construction of D by Algorithm 2. It ensures that the pure mixture effects are orthogonal to order effects in the additive model. From Lemma 1, it is clear that if a simplex design is augmented with additional columns to accommodate a model matrix for a simplex design, then resulting simplex model matrix will be orthogonal to Z as written in Lemma 1. This is because the orthogonality arises from the use of replication. For more details, see the proof in Appendix 3.A (specifically, the use of the replication matrix R).

Lemma 2 (Partition of Sums of Squares)**.** *Suppose D is constructed using Algorithm 2. Let P_D be the projection matrix onto the column space of D. Then for a vector $y \in R^{N\times 1}$, it follows that $y^T P_D y = y^T P_X y + y^T P_Z y$.*

Lemma 2 follows from basic linear model theory. It allows for the use of ANOVA to test for both the effect of the mixture on the response and the order on the response in Model (3.1). In the case of Model (3.4), which has interaction effects, it is still possible to use an ANOVA-type procedure that is similar to the additive case. Let M be the model matrix for Model (3.4). We have the following result:

Lemma 3 (Orthogonality of Mixture and PWO in Model Matrix)**.** *Suppose D is constructed using Algorithm 2. Let M be the model matrix resulting from modifying to columns of D to fit Model (3.4) or (3.5). Then $M = (X, Z)$ such that X is a matrix with columns corresponding to the m mixture components and their 2-way interactions, Z is a matrix with columns corresponding to the first order interactions of the $\binom{m}{2}$ PWO variables with the mixture proportions, and the main PWO effects in the case of Model (3.5), and $X^T Z = \mathbf{0}$.*

Lemma 3 is the extension of Lemma 1 to the more general case of model matrices. As such, it covers models with mixture-order interactions. Thus, if Model (3.4) is fit, the variation can still be separated into two components: variation due to pure mixture terms, and variation involving order from interactions between mixture components and order.

3.6 Examples

3.6.1 Fish Patty Dataset

This example is based on the classical fish patty dataset in Cornell (1990). In this dataset, the response y is the texture of the fish patties, which is measured in grams of force required to puncture the surface of a patty. The fish patties were mixtures of three components: mullet (x_1), sheepshead (x_2), and croaker (x_3). The original dataset included three process variables: temperature (z_1), oven time (z_2), and frying time (z_3). Each of the process variables were coded to have levels $\{-1, 1\}$. The original design was a simplex-centroid design that was run for each of the 2^3 combinations of the three process variables.

Table 3.2: Original Data from Cornell's Fish Patty Experiment (1990)

Process			Mixtures						
z_1	z_2	z_3	Pure 1	Pure 2	Pure 3	12 Blend	13 Blend	14 Blend	Centroid
−1	−1	−1	1.84	0.67	1.51	1.29	1.42	1.16	1.59
1	−1	−1	2.86	1.10	1.60	1.53	1.81	1.50	1.68
−1	1	−1	3.01	1.21	2.32	1.93	2.57	1.83	1.94
1	1	−1	4.13	1.67	2.57	2.26	3.15	2.22	2.60
−1	−1	1	1.65	0.58	1.21	1.18	1.45	1.07	1.41
1	−1	1	2.32	0.97	2.12	1.45	1.93	1.28	1.54
−1	1	1	3.04	1.16	2.00	1.85	2.39	1.60	2.05
1	1	1	4.13	1.30	2.75	2.06	2.82	2.10	2.32

The original dataset is provided in Table 3.2. In this table, the pure mixtures only contain one component, e.g., "Pure 1" corresponds to the point $(x_1, x_2, x_3) = (1, 0, 0)$. In this case, since $m = 3$, the centroid is $(x_1, x_2, x_3) = (1/3, 1/3, 1/3)$. The blend mixtures use half of one component, and half of another. For example, "12 Blend" corresponds to the point $(x_1, x_2, x_3) = (0.5, 0.5, 0)$.

To illustrate the use of the OofA models given in Section 3.4, the process variables z_1, z_2, and z_3 were replaced with the modified PWO variables z_{12}, z_{13}, and z_{23}, respectively. This is simply for the sake of illustration of the methods. In the case where certain components were not used, the corresponding PWO variables were set to 0. There were two rows where the PWO indicator variables represented an impossible ordering, e.g. $(1, -1, 1)$; these two rows (51 and 54) were removed from the dataset.

In the original problem, objectives of the analysis included studying how the process variables impacted the response, as well as the mixture proportions. Additionally, it was desirable for the texture of the fish patties to be between 2.0 and 3.5. Similar objectives are considered in this modified problem. Aims of this analysis will be to see whether the OofA of the mixture components (and the mixture proportions) impacts the response and to see how both OofA and mixture proportions affect the response. Finally, a target value of $T = 2.75$ (the midpoint of 2.0 and 3.5) was considered as an "optimal" texture value. Initially the additive model (3.1) was fit to the data, and an ANOVA was used to determine if the mixture or the order had a significant effect on the response. These results are summarized in Table 3.3. Using a significance level of $\alpha = 0.05$, it is clear that both the mixture proportions and the order of addition have a significant effect on the texture of the fish patties.

Table 3.3: Overall ANOVA Results for the OofA Mixture Model

	Source	SS	df	MS	Fstat	pvalue
1	Mixture	211.9692	6	35.3282	134.2938	4.9052e-27
2	Order	3.4716	3	1.1572	4.3989	8.4958e-03
3	Error	11.838	45	0.2631		

Model (3.1) was compared to Model (3.6), which is provided below.

$$\eta(\mathbf{x}) = \sum_{i=1}^{3} \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j, \quad g(\mathbf{x}, \mathbf{z}) = \sum_{k<l} \delta_{kl} z_{kl} + \gamma_{12}^{1} x_1 z_{12} + \gamma_{23}^{2} x_2 z_{23} + \gamma_{13}^{3} x_3 z_{13} \tag{3.6}$$

We note that the omission of the mixture-order interactions $x_1 z_{13}, x_2 z_{12}$, and $x_3 z_{23}$ in Model (3.6) ensures identifiability; otherwise, we would see that $x_j z_{jk} = x_k z_{jk}$ and the model matrix would not be full rank. The parameter estimates for Models (3.1) and (3.6) are given in Table 3.4.

Table 3.4: Parameter Estimates for the OofA Mixture Models.

	Model (3.1)			Model (3.6)		
Term	Estimate	Std. Error	t value	Estimate	Std. Error	t value
x_1	2.8630	0.1808	15.838	2.8630	0.1863	15.3636
x_2	1.0730	0.1808	5.936	1.0730	0.1863	5.7579
x_3	2.0005	0.1808	11.067	2.0005	0.1863	10.7351
z_{12}	0.1030	0.1405	0.733	0.0075	0.8768	0.0086
z_{13}	0.4726	0.1405	3.365	0.0900	0.8768	0.1027
z_{23}	−0.1064	0.1405	−0.757	−0.1800	0.8768	−0.2053
x_1x_2	−0.9444	0.8405	−1.124	−0.9444	0.8665	−1.0900
x_1x_3	−0.8044	0.8405	−0.957	−0.8044	0.8665	−0.9284
x_2x_3	0.3856	0.8405	0.459	0.3856	0.8665	0.4450
x_1z_{12}				0.2475	1.9427	0.1274
x_2z_{23}				0.1950	1.9427	0.1004
x_3z_{13}				0.9000	1.9427	0.4633

Figure 3.1 shows side-by-side contour plots for Models (3.1) and (3.6), respectively. We note that the shape of the contours for the right panel is different from the left panel due to the interaction terms that are included in Model (3.6). This implies that when the interaction model is used, there is potential for optimal points to lie in different regions. In Figure 3.2, the same models are compared, but the order is changed to $(1, 3, 2)$. The shape of the contours in each plot still differs in terms of gaps between the contours. In all of these cases, the contours have an upward trend when moving towards the right side of the simplex. Additionally, comparing the right panels in Figures 3.1 and 3.2 shows that when the order of addition changes, the shape of the contours change as well. However, a nested F-test between Models (3.1) and (3.6) revealed that the additional

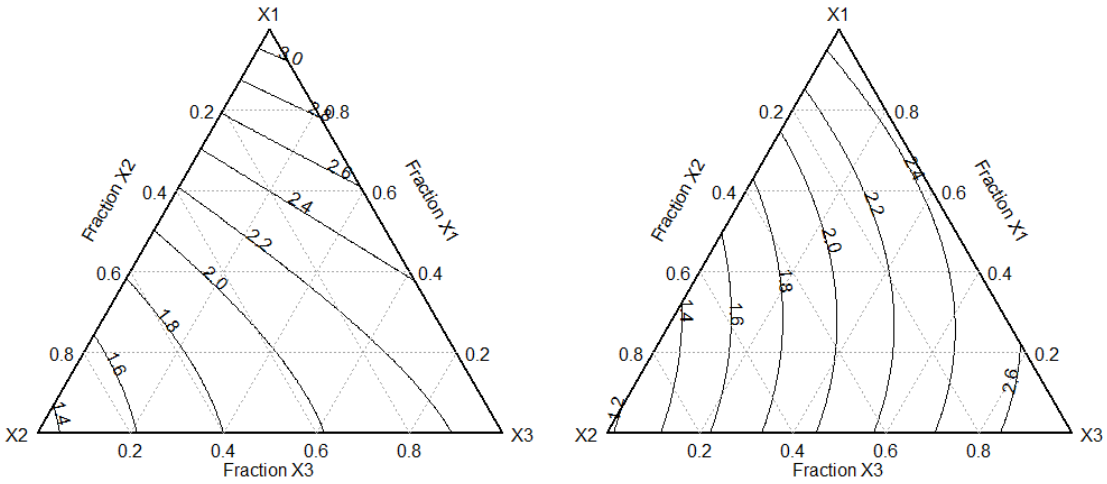


Figure 3.1: Comparison of Models With and Without Interaction Terms for ordering $(2, 1, 3)$. Left: Contour plot for Model (3.1), which has no mixture-order interaction terms. Right: Contour plot for Model (3.6), which has mixture-order interaction terms.

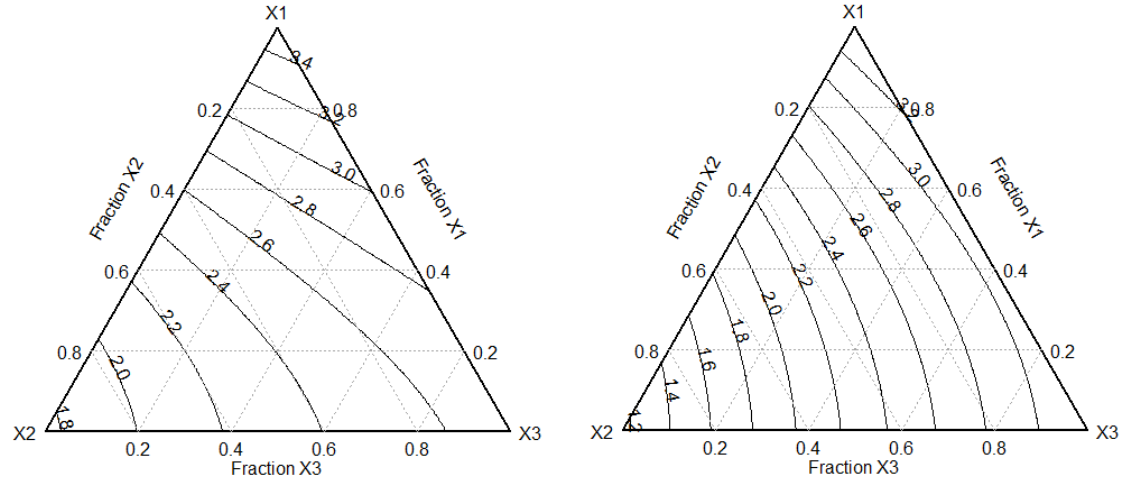


Figure 3.2: Comparison of Models With and Without Interaction Terms for ordering $(1, 3, 2)$. Left: Contour plot for Model (3.1), which has no mixture-order interaction terms. Right: Contour plot for Model (3.6), which has mixture-order interaction terms.

interaction terms in Model (3.6) were not statistically significant (p-value = 0.9509). This does not mean that the interaction terms are unimportant, as they may have important practical interpretations. Furthermore, other interaction models may have a better fit. For example, if we fit Model (3.6) without the OofA main effect terms $\gamma_{kl} z_{kl}$, then the resulting model has lower AIC (90.92 vs 91.29,96.85) and BIC (110.8 vs 111.18,122.71)

than Models (3.1) and (3.6), respectively.

By examining all possible orders, we found the mixture and order settings that gave the closest patty texture to the target of $T = 2.75$ for both models. For both models, there were multiple combinations of the mixture and order that gave a predicted response nearly equal to T (within 10^{-10}). These are summarized in Table 3.5. It should be noted that for the ordering $(2, 1, 3)$, the value of $\mathbf{x}^*$ for model (3.1) is quite different than it is under Model (3.6). Careful consideration should be taken when deciding which of these points to use. For example, it may not be practical to have a mixture proportion as low as 0.003, so one should carefully consider if the first optimal point for Model (3.6) is a reasonable choice.

Table 3.5: Mixture proportions and orderings with a response matching the target of $T = 2.75$.

Model (3.1)		Model (3.6)	
$\mathbf{x}^*$	$\mathbf{a}^*$	$\mathbf{x}^*$	$\mathbf{a}^*$
(0.603,0.054,0.343)	(1,2,3)	(0.611,0.003,0.385)	(1,2,3)
(0.469,0.161,0.370)	(1,3,2)	(0.399, 0.167, 0.434)	(1,3,2)
(0.767,0.033,0.200)	(2,1,3)	(0.838, 0.078, 0.083)	(3,1,2)
		(0.026, 0.020, 0.954)	(2,1,3)

The purpose of this example is to both demonstrate how OofA Mixture models can be used and also to show the impact of including interaction terms in an OofA Mixture model. The inclusion of interaction terms in this example altered the shape of the contours. This resulted in different optimal points for the models with and without interaction when the objective was hitting a target response value.

3.6.2 Chocolate Manufacturing Dataset

In this section, data from Aidoo et al. (2014) are used. This was a study on the use of inulin and polydextrose mixtures as a replacement for sucrose in the manufacturing process of sugar-free chocolate. There were $m = 2$ mixture components, which were inulin (x_1) and polydextrose (x_2). One of the responses examined in this study was casson viscosity, which was reported as an average over 3 repeated measurements. It is desirable to minimize the mean casson visocity. Originally, there was no explicit mention of the order of addition of the two mixture components. It will be shown that assuming different mixing orders for the inulin and polydextrose in the trials can yield different models and optimal responses. In the original data, each binary blend was replicated twice, which implies that one order can be assigned to each binary blend. Consider two

assignments of the mixing orders, denoted by $z_{12}^{(1)}$ and $z_{12}^{(2)}$ and shown in Table 3.6. Aidoo et al. (2014) obtained the following linear model for the response of casson viscosity:

$$\hat{y} = 4.728x_1 + 2.126x_2 \tag{3.7}$$

Table 3.6: Data from Aidoo et al. (2014) augmented with two Mixing Orders $z_{12}^{(1)}$, $z_{12}^{(2)}$

Inulin (x_1)	Polydextrose (x_2)	Casson Viscosity (y)	$z_{12}^{(1)}$	$z_{12}^{(2)}$
0.25	0.75	2.55	1	1
0.50	0.50	4.37	1	−1
0.75	0.25	3.73	−1	1
1.00	0.00	4.72	0	0
0.00	1.00	2.09	0	0
0.50	0.50	3.29	−1	1
1.00	0.00	4.56	0	0
0.25	0.75	2.29	−1	−1
0.75	0.25	4.36	1	−1
0.00	1.00	2.31	0	0

We compare model (3.7) with a model of the following form:

$$y = \beta_1 x_1 + \beta_2 x_2 + \delta_{12} z_{12} + \epsilon \tag{3.8}$$

Table 3.7: ANOVA for Mixing Orders $z_{12}^{(1)}, z_{12}^{(2)}$

Orders	Source	SS	df	MS	F	p-value
$z_{12}^{(1)}$	Mixture	125.9063	2	62.9531	541.9065	< 0.0001
	Order	0.6468	1	0.6468	5.5679	0.0503
	Error	0.8132	7	0.1162		
$z_{12}^{(2)}$	Mixture	125.9063	2	62.9531	397.1491	< 0.0001
	Order	0.3504	1	0.3504	2.2107	0.181
	Error	1.1096	7	0.1585		

We first check whether the order of addition is significant for each mixing order configuration z_{12}^1 and z_{12}^2. This can be done by fitting model (3.8) to the data, partitioning the model sum of squares into the sum of squares due to mixture and sum of squares due to order (as shown in Lemma 2), and then using ANOVA. Table 3.7 shows the ANOVA for each of the two mixing order configurations. For the first mixing orders, the order is significant at the $\alpha = 0.10$ level, while it is not significant for the second. This demonstrates that the assumption of mixing order does have an impact on the statistical analysis. Using model (3.8), the optimal settings are found to be $(x_1^*, x_2^*) = (0.001, 0.999)$, and adding inulin before polydextrose to achieve a minimum of $y^* = 1.80$. If Model (3.7)

is fit, the optimal (x_1^*, x_2^*) are the same, but a minimal casson viscosity is $y^* = 2.126$, which is not as small as the one obtained by considering the order of addition.

In this section, the models proposed in Section 3.4 were easy to fit once an addition order was assumed. This was because the design used had replicated each mixture the appropriate number of times. Moreover, the design from Section 3.3 made it simple to determine whether the order of addition has a significant effect on the response. This illustrates that when designing an OofA Mixture experiment, some of the new concerns are using the appropriate number of replicates and assigning addition orders to each replicate in advance.

3.7 Simulation Study

Simulations were conducted to compare the optimal mixture proportions produced by the pure mixture model (2.5) to those produced by the interaction model (3.4) under various scenarios. The aim was to maximize the response. In the simulations, the optimization of the response was done in R using the package Rsolnp; see Ghalanos and Theussl (2012). Two settings (denoted Mixture-1, Mixture-2) were used to generate the parameters for the response surface, and another two settings (denoted Order-1, Order-2) were used to generate the parameters for the interaction effects in Model (3.4). Mixture data was simulated with $m = 3, 4, 5$ components under Model (3.4). All of the scenarios used normal errors with constant variance σ^2. Simulations were run for $\sigma^2 = 0.05$ and 1. In all cases the optimal ordering was to place the components in ascending order.

- **Mixture-1 (Edge):** The optimal point lies on the edge of the simplex between x_1 and x_2. In this case, the parameters for the response surface were $\beta_1 = 4, \beta_2 = 2$, $\beta_i = 1$ for $i = 3, 4, \ldots, m$, and $\beta_{ij} = \beta_i \beta_j$ for $i < j$. For example, when $m = 3$, $\beta = (3, 3, 1, 9, 3, 3)$.

- **Mixture-2 (Center):** The optimal point lies near the center of the simplex. In this case, the parameters for the response surface were $\beta_i = 3$ for $i = 1, \ldots, m$ and $\beta_{ij} = \beta_i \beta_j - i + j$. When $m = 3$, $\beta = (3, 3, 3, 7, 8, 7)$.

- **Order-1 (Constant):** "Constant" means that the interaction terms were the same for each $kl \in \mathcal{P}$ across i, i.e. $\gamma_{kl}^i = \gamma_{kl}$. The interaction effects are $\gamma_{12} = 1, \gamma_{13} = 2, \ldots, \gamma_{m-1,m} = \binom{m}{2}$, e.g. for $m = 3$, $\gamma = (1, 2, 3, 1, 2, 3, 1, 2, 3)$.

- **Order-2 (Varying):** "Varying," means that the interaction terms were not constant across i for each $kl \in \mathcal{P}$. In this case, $\gamma_{kl}^{i} = i/1, i/2, \ldots, i/\binom{m}{2}$. For example, when $m = 3$, $\gamma = (1, 1/2, 1/3, 2, 1, 1/3, 3, 3/2, 1)$.

In these simulations, a number of settings were varied, with the underlying goal of allowing more generalized conclusions to be made about the quality of the optimal proportions found the mixture-only model (2.5) as opposed to those found using the interaction model (3.4). These settings include the error variance (small vs large), location of the optimal point in the simplex (edge vs center), nature of order effect parameters (constant vs varying), and the number of mixture components m. In the Mixture-1 setting, the optimal point is taken to be on the edge between x_1 and x_2, without loss of generality. On the other hand, the Mixture-2 setting represents the case where the optimal point is near the center of the simplex. The Order-1 setting is established so that the interaction effects are the same for each $kl \in \mathcal{P}$, mimicking an additive effect. Otherwise, in the the Order-2 setting, these interaction effects are allowed to vary.

Table 3.8 shows the optimal mixture proportions $\mathbf{x}^*$, optimal ordering, and maximum response y^* for $m = 3$ and $\sigma^2 = 0.05$. The remaining cases are shown in Appendix 3.B. In all cases, the optimal proportions found using Model (2.5) differed from those found using Model (3.4). For example, when $m = 3$, in the case of Mixture-1 and Order-2, Model (2.5) places the optimal point on the edge of the simplex, while Model (3.4) places the optimal point closer to the interior of the simplex. This case is shown in Figure 3.3.

Table 3.8: Optimal Proportions, Order, and Response from Simulation, $m = 3$, $\sigma^2 = 0.05$

m	Mixture	Order	Model	$\mathbf{x}^*$	$\mathbf{a}^*$	y^*
3	1	1	(2.5)	(0.658,0.342,0)	NA	5.076
			(3.4)	(0.754,0.211,0.036)	(1,2,3)	10.719
		2	(2.5)	(0.645,0.355,0)	NA	5.293
			(3.4)	(0.436,0.373,0.191)	(1,2,3)	7.729
	2	1	(2.5)	(0.341,0.284,0.375)	NA	6.550
			(3.4)	(0.276,0.294,0.43)	(1,2,3)	12.61
		2	(2.5)	(0.333,0.316,0.351)	NA	6.345
			(3.4)	(0.081,0.410,0.509)	(1,2,3)	10.512

In general, the simulations show that when the order of addition of the mixture components has an effect on the response, then the optimal mixture proportions $\mathbf{x}^*$ and optimal response y^* found by the traditional simplex model may be misleading. In particular, it is found that if there are significant mixture-order interactions, then the optimal proportions $\mathbf{x}^*$ occur at different locations in the simplex than in traditional models. We recommend to first fit Model (3.1), which has no interaction effects, to

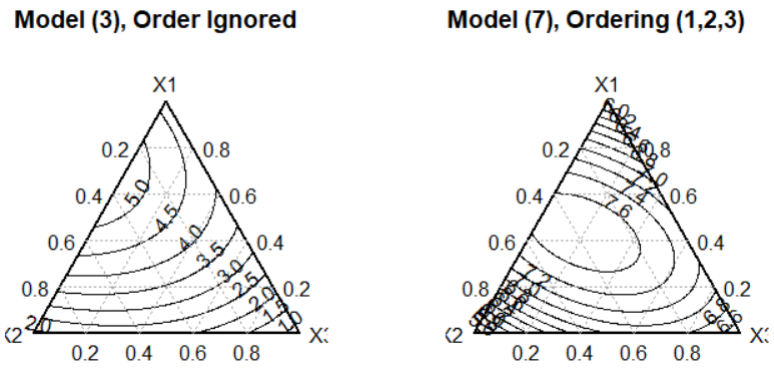


Figure 3.3: Comparison of Models, $m = 3$, Mixture Setting 1, Order Setting 2. Left: The response surface using Model (2.5). Right: The response surface using Model (3.4).

determine if the order of addition has any effect on the response. This can be followed up by a comparison of this additive model and one of the interaction models, e.g. Model (3.5). The purpose of this comparison is to determine if there are any significant mixture-order interactions. In Section 3.8, the importance of checking for interactions on searching for global optima be discussed in some more detail.

3.8 Identifying Optimal Mixture and Order For Large m

The overall goal of an OofA Mixture experiment is to identify the mixture proportions and order that produce an optimal response. The first step towards achieving this goal is fitting an OofA Mixture model of the general form (3.2) to the set of design points enumerated in Section 3.3. Once a model has been fit, it can be used to find an optimal response. This section focuses on this sub-problem. The inputs to model (3.2) are an addition order $\mathbf{a}$ and a vector of mixture proportions $\mathbf{x}$. These inputs exist the Cartesian product space $\mathcal{A} \times \mathcal{S}$. We emphasize that this space is a mixture of discrete and continuous spaces. In particular, the values in $\mathcal{A}$ must be valid permutations of $(1, 2, \ldots, m)$. This problem is an instance of the Mixed Integer Programming (MIP) problem, as we are trying to optimize some function $f(\mathbf{x}, \mathbf{a})$ over a set where the inputs $\mathbf{a}$ are represented by integer variables, and $\mathbf{x}$ are continuous. As such, optimization methods that treat all predictors as continuous are not ideal for this scenario.

In Sections 3.6 and 3.7, a "brute-force" algorithm was used to identify the optimal mixture and order settings. This is detailed below in Algorithm 3, which searches for the optimal proportions and order $\mathbf{x}^*, \mathbf{a}^*$ that produce a minimized response y^*.

Algorithm 3: Brute Force: Find Optimal (Minimum) $y^*, \mathbf{x}^*, \mathbf{a}^*$

Input: A fitted model $\hat{\eta}(\mathbf{x}) + \hat{g}(\mathbf{x}, \mathbf{a})$

for *all possible orderings* $\boldsymbol{a}$ **do**

1. $\mathbf{x}^*_{\mathbf{a}} = \arg\min_{\mathbf{x}} \hat{\eta}(\mathbf{x}) + \hat{g}(\mathbf{x}, \mathbf{a})$ subject to $\sum_i \mathbf{x}_i = 1$
2. $y^*_a = \hat{\eta}(\mathbf{x}^*_{\mathbf{a}}) + \hat{g}(\mathbf{x}^*_{\mathbf{a}}, \mathbf{a})$

end

3. $\mathbf{a}^* = \arg\min_a y^*_a$
4. $\mathbf{x}^* = \mathbf{x}^*_{\mathbf{a}^*}$
5. $y^* = y^*_{a^*}$

return $\mathbf{x}^*, \mathbf{a}^*, y^*$

As inputs, Algorithm 3 uses a fitted Mixture OofA model, e.g. Model (3.1) or (3.4). It loops through all possible permutations of $(1, 2, \ldots, m)$. In Step 1, Algorithm 3 finds the optimal mixture proportions $\mathbf{x}^*_{\mathbf{a}}$ for a fixed permutation $\mathbf{a}$. In our R implementation, the package Rsolnp Ghalanos and Theussl (2012) was used to perform this constrained optimization. The estimated responses $y^*_{\mathbf{a}}$ are recorded in Step 2. In Step 3, the order corresponding to the largest response is selected, and the corresponding mixture proportions are extracted in Step 4. To change this to maximization, one only needs to modify steps 1 and 3. Algorithm 3 is computationally feasible and efficient when m is small. However, for large m (e.g. $m \geq 10$), there are millions of possible orders to consider, and looping through all possible orders is inefficient in terms of CPU time.

3.8.1 Optimization Under Additive Model

When choosing an approach for finding the optimal mixture and order, an initial concern is determining if there are significant mixture-order interactions. If the model is purely additive (i.e. no interactions between mixture and order), then the general form of Model (3.2) simplifies to

$$f(\mathbf{x}, \mathbf{a}) = \eta(\mathbf{x}) + g(\mathbf{a}) \tag{3.9}$$

The simplification in Model (3.9) occurs because, in this case, $g(\mathbf{x}, \mathbf{a})$ only depends on the order of the components, and not their mixture proportions. Under Model (3.9), the optimal mixture and order can be found by performing two independent optimizations.

$$\arg\min_{\mathbf{x} \in \mathcal{S}} f(\mathbf{x}, \mathbf{a}) = \arg\min_{\mathbf{x} \in \mathcal{S}} \eta(\mathbf{x}) \tag{3.10}$$

$$\arg\min_{\mathbf{a}\in\mathcal{A}} f(\mathbf{x},\mathbf{a}) = \arg\min_{\mathbf{a}\in\mathcal{A}} g(\mathbf{a}) \tag{3.11}$$

The optimization in (3.10) shows that the optimal mixture does not depend on the order. Likewise, the optimization in (3.11) shows that the optimal order does not depend on the mixture proportions. Therefore, finding the optimal mixture and order under the additive model is computationally simpler than in the case of mixture-order interactions. To determine if mixture-order interaction is significant, one can also perform a nested F-test between the additive model (3.1) and Model (3.5), which allows for PWO main effects and interactions. As seen in Section 3.6, criterion such as AIC, BIC can be used to perform this model selection. The optimization in (3.10) is over a continuous space, and the typical estimated form of $\eta(\mathbf{x})$ is a quadratic function. Therefore, a variety of existing software packages can be used to minimize the estimated response surface subject to the constraint that $\sum_i x_i = 1$. For example, the R package Ghalanos and Theussl (2012) was employed in Section 3.7 for this purpose.

To identify the optimal order of addition for large m, a topological sorting method from Zhao et al. (2021) is employed. When a model of the form (3.9) is fit, the significant PWO coefficients from $\hat{g}(\mathbf{a})$ are extracted, e.g. the estimated coefficients $\hat{\delta}_{jk}$ that have a p-value less than or equal to $\alpha = 0.05$. These coefficients are used to form a Directed Acylic Graph (DAG), where the vertices are $(1, 2, \ldots, m)$. Suppose we are minimizing the response. Then, for each significant $\hat{\delta}_{jk}$, form a directed edge from j to k in the DAG if $\hat{\delta}_{jk} < 0$; otherwise, add a directed edge from k to j. Then, the set of candidate optimal orders is taken to be the set of all possible topological sorts of this DAG, i.e., if j has a directed edge to k, then j precedes k in the ordering. Fortunately, many efficient algorithms for identifying topological sorts of a DAG exist. We utilize the Rfast implementation of topological sort (Papadakis et al., 2022).

As an example of optimization in the additive model case, we consider the case where the design is the $\{8, \ell\}$ OofA-SLD. The responses y are generated from a modified version of the Levy function (Laguna and Marti, 2005), plus $N(0, 0.01^2)$ random noise. The Levy function was chosen for the response surface because it is a challenging test function with many local optima, and it is adaptable to m dimensions. The general form of the Levy function is given in Equation (3.12), which was used to simulate the responses.

$$f(\mathbf{x},\mathbf{a}) = \sin^2(\pi w_1) + \sum_{i=1}^{m}(w_i - 1)^2[1 + 10\sin^2(\pi w_i + 1)] + \tag{3.12}$$

$$(w_m - 1)^2[1 + \sin^2(2\pi w_d)] + \sum_{j<k}(z_{jk}(\mathbf{a}) - 1)^2$$

$$w_i = 1 + \frac{x_i - 1/m}{4} \quad i = 1, \ldots, m$$

The modified Levy function (3.12) has many local minima, but it is not difficult to see that it has a global minimum of $y^* = 0$ at $\mathbf{x}^* = (1/m, \ldots, 1/m)$ and $\mathbf{a}^* = (1, 2, \ldots, m)$. When $x_i = 1/m$, then $w_i = 1$, and each $(w_i - 1)^2 = 0$ for $i = 1, 2, \ldots, m$. If $\mathbf{a} = (1, 2, \ldots, m)$, then $z_{jk}(\mathbf{a}) = 1$ for all pairs jk, and the summation $\sum_{jk}(z_{jk}(\mathbf{a}) - 1)^2$ is minimized at zero. By construction, this function does not have mixture-order interactions.

Table 3.9: Predicted Optimal Mixture and Order for Simulated Levy Response, $m = 8$

Simulation Settings		Predicted $\mathbf{x}^*$	Predicted $\mathbf{a}^*$	Correct?
$\ell = 3$	$\sigma^2 = 0.001^2$	(0.125,0.125,...,0.125)	(1,2,3,4,5,6,7,8)	Yes
	$\sigma^2 = 0.01^2$	(0.125,0.125,...,0.125)	(1,2,3,4,5,6,7,8)	Yes
	$\sigma^2 = 0.1^2$	(0,0,0,0,0,1,0,0,0)	(1,2,3,4,5,6,7,8)	No
$\ell = 5$	$\sigma^2 = 0.001^2$	(0.125,0.125,...,0.125)	(1,2,3,4,5,6,7,8)	Yes
	$\sigma^2 = 0.01^2$	(0.125,0.125,...,0.125)	(1,2,3,4,5,6,7,8)	Yes
	$\sigma^2 = 0.1^2$	(0.125,0.125,...,0.125)	(1,2,3,4,5,6,7,8)	Yes

Table 3.9 shows the simulation results for this example. The first two columns show the degree ℓ of the OofA SLD used, as well as the variance σ^2 of the simulated errors. The standard deviation σ was increased by a power of 10 at each step to simulate an increase in noise. In every case, each $\hat{\delta}_{jk}$ was significant at the $\alpha = 0.05$ level. Therefore, the DAG in Figure 3.4 was formed, with a directed edge from j to k for every $j < k$. The only valid topological sort identified was $(1, 2, \ldots, m)$, which is the correct answer.

The optimal mixture proportions were found in every case except for when $\ell = 3, \sigma^2 = 0.1$. In this case, the correct addition order is identified, but the increase in noise causes the the predicted $\mathbf{x}^*$ to end up on the border, i.e., trapped at a local optima (of which the Levy function has many). This is remedied by increasing the degree of the design from $\ell = 3$ to $\ell = 5$, as more design points make this prediction more stable. In this case, even with a function that provides a numerically challenging mixture response surface, our methods can be used to identify optimal mixture proportions and order of addition.

If the objective is to minimize the distance between the response $f(\mathbf{x}, \mathbf{a})$ and a given target T, then the independent optimizations in (3.10), (3.11) may not lead to the optimal answer. This is because, under the additive model, the objective function can be written

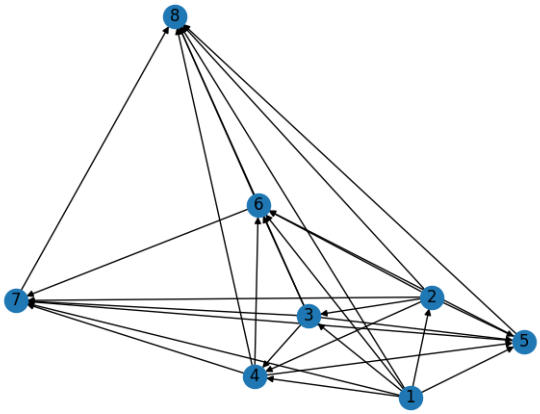


Figure 3.4: Directed Acyclyic Graph (DAG) for Finding Optimal Order

as follows:

$$(f(\mathbf{x},\mathbf{a})-T)^2=(\eta(\mathbf{x})+g(\mathbf{a})-T)^2=(\eta(\mathbf{x})+g(\mathbf{a}))^2-2T(\eta(\mathbf{x})+g(\mathbf{a}))+T^2$$

The quadratic term $(\eta(\mathbf{x})+g(\mathbf{a}))^2$, when expanded, contains $2\eta(\mathbf{x})g(\mathbf{a})$, which has terms that are akin to mixture-order interactions. Therefore, the methods in the following section are recommended for large m if the goal is hitting an optimal target, regardless of which model is preferred.

3.8.2 Optimization with Mixture-Order Interactions

As demonstrated in Section 3.7, the presence of mixture-order interactions alters the shape of the response surface, and, by extension, the location of global minima. Therefore, the independent optimizations in (3.10), (3.11) are no longer guaranteed to yield the best results. Additionally, for large m, the brute force approach is not ideal. A method for finding the optimal mixture and order should efficiently search through the space $\mathcal{S}\times\mathcal{A}$ in a way that avoids becoming trapped at local optima.

As an alternative to a brute force approach, we propose a Simulated Annealing (SA) algorithm. Simulated Annealing is a probabilistic approach that can be used to efficiently search for the global minimum (or maximum) of a function (Bertsimas and Tsitsiklis, 1993). The basic idea behind this algorithm is to allow for some sub-optimal changes in the objective function, but have the probability of these changes decrease the longer the algorithm runs. This algorithm is particularly useful for cases where the objective function has many local minima. Fitted models of the form (3.2) will likely exhibit local

minima in $\mathcal{S}$ for each fixed $\mathbf{a} \in \mathcal{A}$. Hence, this method is an appealing approach for our purposes. The implementation of Simulated Annealing is given below in Algorithm 4.

Algorithm 4: Simulated Annealing: Find Optimal (Minimum) $y^*, \mathbf{x}^*, \mathbf{a}^*$

Input: A fitted model $\hat{f}(\mathbf{x}, \mathbf{a}) = \hat{\eta}(\mathbf{x}) + \hat{g}(\mathbf{x}, \mathbf{a})$, initial guess $(\mathbf{x}_0, \mathbf{a}_0)$, number of iterations n_i, small real $\epsilon > 0$

1. Initialize $\mathbf{x}^*, \mathbf{a}^*, y^*$.

for $t = 1, 2, \ldots, n_i$ **do**

2. Let $\mathbf{x}_0, \mathbf{a}_0$ be the current solution. Set $\mathbf{x}_1 = \mathbf{x}_0, \mathbf{a}_1 = \mathbf{a}_0$.
3. Simulate $U_1 \sim U[0, 1]$.
if $U_1 < 0.5$ **then**
4. $\mathbf{a}_1 = \text{random\_swap}(\mathbf{a}_0)$
else
5. $\mathbf{x}_1 = \text{random\_neighbor}(\mathbf{x}_0, \epsilon)$
end
6. Simulate $U_2 \sim U[0, 1]$.
if $\hat{f}(\boldsymbol{x}_1, \boldsymbol{a}_1) < \hat{f}(\boldsymbol{x}_0, \boldsymbol{a}_0)$ **then**
7. $\mathbf{x}_0 = \mathbf{x}_1, \mathbf{a}_0 = \mathbf{a}_1$
8. If $\mathbf{x}_0, \mathbf{a}_0$ is a better solution, update $\mathbf{x}^* = \mathbf{x}_0, \mathbf{a}^* = \mathbf{a}_0, y^* = \hat{f}(\mathbf{x}_0, \mathbf{a}_0)$.
else if $U_2 < \exp\left((-(\hat{f}(\boldsymbol{x}_1, \boldsymbol{a}_1) - \hat{f}(\boldsymbol{x}_0, \boldsymbol{a}_0))/(1/\log(t+1))\right)$ **then**
9. $\mathbf{x}_0 = \mathbf{x}_1, \mathbf{a}_0 = \mathbf{a}_1$
10. If $\mathbf{x}_0, \mathbf{a}_0$ is a better solution, update $\mathbf{x}^* = \mathbf{x}_0, \mathbf{a}^* = \mathbf{a}_0, y^* = \hat{f}(\mathbf{x}_0, \mathbf{a}_0)$.

end

return $\mathbf{x}^*, \mathbf{a}^*, y^*$

Algorithm 4 takes as inputs a fitted model $\hat{f}$, initial mixture proportions $\mathbf{x}_0$, initial order $\mathbf{a}_0$, a (large) number of iterations n_i, and a small real positive ϵ. During each iteration of the algorithm, a solution that is close to the current solution is randomly selected. With probability 0.5, the mixture proportions are left unchanged, but the order is changed. Otherwise, the order is left unchanged, but the mixture proportions are changed. In line 4, the function random_swap takes a permutation $\mathbf{a}_0$ and samples an index i from the range $(1, 2, \ldots, m-1)$ with uniform probability. It then returns a new permutation that is identical to $\mathbf{a}_0$, apart from having $\mathbf{a}_0[i]$ exchanged with $\mathbf{a}_0[i+1]$. In line 5, the function random_neighbor takes a vector of m proportions $\mathbf{x}_0$. It randomly samples one index i from the range $(1, 2, \ldots, m)$. Then, it creates a new vector $\mathbf{x}_1$, where $\mathbf{x}_{1i} = \mathbf{x}_{0i} + \epsilon$ and $\mathbf{x}_{1j} = \mathbf{x}_{0j} - \mathbf{x}_{0j}\epsilon/(1 - \mathbf{x}_{1i})$ for all $j \neq i$. Essentially, this makes a small change in the mixture proportions while still ensuring that they sum to 1. If such a change would make $\mathbf{x}_{1i} > 1$, then a new component is selected. If the proposed solution $(\mathbf{x}_1, \mathbf{a}_1)$ is

better than the current solution, then it is accepted in line 7. Then, the proposed solution is checked against the current best solution $(\mathbf{x}^*, \mathbf{a}^*)$, and the best solution is updated accordingly in line 8. Otherwise, if the proposed solution is worse than the current solution, it is accepted with probability $\exp\left(-(\hat{f}(\mathbf{x}_1, \mathbf{a}_1) - \hat{f}(\mathbf{x}_0, \mathbf{a}_0))/(1/\log(t+1))\right)$. This probability will decrease as t increases, which encourages sub-optimal updates to occur less often later on in the algorithm. If the proposed solution is much larger than the current solution, the acceptance probability will be very low, as $\exp(-x) \to 0$ as $x \to \infty$. The term $1/\log(t+1)$ can be generalized to any "cooling schedule" $T(t)$ which is a non-increasing, positive function that satisfies $\lim_{t\to\infty} T(t) = 0$ (Bertsimas and Tsitsiklis, 1993). This may be changed by the user, but we found that using a cooling schedule of the form $T(t) = c/\log(t+1)$ for constant c worked well in this application.

As an example, consider the case where $m = 8$. We generate the responses y using a modified version of the Sphere function (Laguna and Marti, 2005). This function is given by

$$f(\mathbf{x}, \mathbf{a}) = \sum_{i=1}^{m} \left(x_i - \frac{1}{m}\right)^2 + \sum_{i=1}^{m-1} \left(x_i z_{i,i+1}(\mathbf{a}) - \frac{1}{m}\right)^2 \tag{3.13}$$

In general, the modified Sphere function (3.13) has a global minimum of $y^* = 0$ at $\mathbf{x}^* = (1/m, \ldots, 1/m)$ and $\mathbf{a}^* = (1, 2, \ldots, m)$. The first summation corresponds to pure mixture effects of the x_i for $i = 1, \ldots, m$. The second summation includes interaction between x_i and the order of proportions x_i and x_{i+1} for $i = 1, 2, \ldots, m-1$. Specifically, if component $i+1$ occurs before component i, then the i^{th} term can no longer go to zero when $x_i = 1/m$; this is how the order for the global minimum is derived.

Mean-zero Gaussian noise was added with increasing levels of variance σ^2. An $\{8, \ell\}$ OofA-SLD was used to fit the full OofA Mixture interaction model (3.2). In each case, it was found that interaction parameters γ^i_{jk} were insignificant if $j, k \neq i, i+1$; this makes sense, considering the generating function (3.13). These terms were removed, and the data were fit to the resulting reduced model.Algorithm 4 was executed for $n_i = 10000$ iterations with cooling schedule $T(t) = 1/\log(1+t)$. Random ϵ-neighbors for mixture proportions were selected using $\epsilon = 0.001$. In general, the performance of the algorithm is not sensitive to the choice of ϵ, provided that $\epsilon < \sqrt{2}/\ell$. This is because $\sqrt{2}/\ell$ is the distance between any pair of points in the SLD. Having a value of ϵ larger than this would encourage erratic jumps larger than the points used to fit the model. The algorithm ran using 10 randomly assigned starting points. The results with the lowest predicted optimal value of y were reported in Table 3.10.

Table 3.10: Predicted Optimal Mixture and Order from Simulated Annealing, $m = 8$

Simulation Settings		$\hat{\mathbf{x}}^*$	$\hat{\mathbf{a}}^*$	$\hat{y}^*$
$\ell = 3$	$\sigma^2 = 0.001^2$	(0.18,0.09,0.14,0.12,0.13,0.09,0.16,0.09)	(1,2,3,4,5,6,7,8)	0.0004
	$\sigma^2 = 0.01^2$	(0.16,0.11,0.13,0.10,0.14,0.10,0.17,0.08)	(1,2,3,4,5,6,7,8)	0.0020
	$\sigma^2 = 0.1^2$	(0.18,0.08,0.19,0.09,0.12,0.11,0.17,0.05)	(1,2,3,4,5,6,7,8)	-0.0472
$\ell = 5$	$\sigma^2 = 0.001^2$	(0.16,0.11,0.15,0.11,0.12,0.13,0.12,0.10)	(1,2,3,4,5,6,7,8)	0.0030
	$\sigma^2 = 0.01^2$	(0.21,0.05,0.18,0.08,0.15,0.12,0.17,0.05)	(1,2,3,4,5,6,7,8)	-0.0350
	$\sigma^2 = 0.1^2$	(0.17,0.11,0.12,0.13,0.12,0.12,0.14,0.08)	(1,2,3,4,5,6,7,8)	0.0078

The true global minimum is $y^* = 0, \mathbf{x}^* = (0.125, \ldots, 0.125), \mathbf{a}^* = (1, 2, 3, 4, 5, 6, 7, 8)$.

As shown in Table 3.10, the optimal order was correctly predicted in each of the six examined cases. In general, as the variance of the errors σ^2 increased, the model estimates became more noisy, which made the estimates of the optimal points less reliable. However, all optimal mixture points fall reasonably close to the centroid of the simplex, with predicted outputs close to the true global minimum of 0. When $\ell = 5$, there are more data points in the design, so the results are slightly more reliable. In an experimental setting, a confirmation run would likely be executed at $\hat{\mathbf{x}}^*, \hat{\mathbf{a}}^*$ to ensure that it is indeed optimal. It would also be wise to sample a number of points in a small radius around $\hat{\mathbf{x}}^*$, if this is feasible in the budget.

3.9 Conclusion

In this chapter, a framework for Order-of-Addition analysis in traditional mixture designs was provided. Full OofA Mixture designs for this experiment were constructed, which ensure that simplex design parameters and pairwise ordering effects are orthogonal. This orthogonality property also extends to mixture-order interaction effects. Two general models were proposed that allow for interaction effects between mixture and order. An identifiability condition was proposed that, if met, is sufficient for the estimation of all parameters in these models. The designs presented in this chapter can be used to determine whether the order and the mixture have a significant effect on the response. Simulations provide empirical evidence which indicates that that significant interactions between order and mixture, may cause the optimal mixture proportions found by traditional models to be misleading. Finally, methodology for finding optimal mixture proportions and addition order when the number of components is large was explored. For the case of mixture-order interactions, a simulated annealing algorithm was proposed to search for the optimal mixture and order. This approach is also useful

for "target is better" criteria; i.e., trying to minimize the distance between the response and a target T.

In this chapter, the first steps for designing OofA Mixture experiments are laid out. The remaining chapters will provide useful extensions of this work. The full designs proposed in this chapter are not ideal for large values of m due to their rapidly increasing run size. In Chapter 4, algorithms will be employed to reduce the run size according to an optimality criterion. More attention will also be given to the extreme vertices design, as it is very realistic that the mixture proportions will have single component constraints $0 \leq L_i \leq x_i \leq U_i \leq 1$ for each $i = 1, \ldots, m$. With constraints on the mixture proportions, the Simplex Lattice and Centroid designs cannot be used as a base design for the methods employed in this chapter. Another integral extension of this work is to the case where there are constraints placed on the set of possible addition orders. In Chapter 5, the case where the chemicals may only sequentially interact with a set of neighbors (i.e. a network structure) is examined through the lens of an OofA experiment.

3.A Appendix: Proofs

Proof of Lemma 1. Let $\mathbf{S}$ be a simplex design with n runs for the m mixture components. Then $\dim(\mathbf{S}) = n \times m$. Suppose that we construct D using Algorithm 1. Then by the algorithm, $X = RS$ where R is a $N \times n$ matrix with entries

$$R_{ki} = \begin{cases} 1 & \text{if row i of } \mathbf{S} \text{ is row k of } \mathbf{X} \\ 0 & \text{otherwise} \end{cases}$$

For convenience, use column vector notation $R = (\mathbf{R_1}, \ldots, \mathbf{R_n})$. As per Algorithm 2, let Z be a $N \times \binom{m}{2}$ matrix entries Z_{kj} that are the modified PWO variables. Write Z in column vector form as $Z = (Z_1, \ldots, Z_{\binom{m}{2}})$. Then, notice that $\mathbf{X^T Z} = \mathbf{S^T R^T Z}$. So, it suffices to prove that $\mathbf{R^T Z} = \mathbf{0}$. This will be shown by proving that $R_i^T Z_j = 0$ for all $i = 1, \ldots, n$ and $j = 1, \ldots, \binom{m}{2}$. Let $\mathcal{K}$ be the set of indices where row k of X is the same as row i of $\mathbf{S}$. Then it follows that

$$R_i^T Z_j = \sum_{k=1}^{N} R_{ki} Z_{kj} = \sum_{k \in \mathcal{K}} Z_{kj}$$

Let column j of Z correspond to the pair $pq \in \mathcal{P}$. Let x be the i^{th} row of $\mathbf{S}$. Let

$$I = \{i \in \{1, \ldots, m\} \mid x_i \neq 0\}$$
$$\mathcal{A} = \{(a_1, \ldots, a_m) \mid (a_i)_{i \in I} \text{ is a permutation of the elements of } I \text{ and if } i \notin I,\ a_i = i\}$$

Then it follows that

$$\sum_{k \in \mathcal{K}} Z_{kj} = \sum_{a \in \mathcal{A}} z_{pq}(x, a) = \begin{cases} 0 & \text{if } x \text{ is a vertex on the simplex} \\ 0 & \text{if } x_p = 0 \text{ or } x_q = 0 \\ \frac{|\mathcal{A}|}{2} - \frac{|\mathcal{A}|}{2} = 0 & \text{otherwise} \end{cases}$$

The first two cases follow directly from the definition of the modified PWO components. The first case is trivial; if x is a vertex on the simplex then only one component of x will be nonzero. If it is the (third) case that $x_p, x_q \neq 0$ then since $\mathcal{A}$ is the set of all permutations of the elements of I, then the number of times we have p appear before q is $\frac{|\mathcal{A}|}{2}$, and the number of times q appears before p is also $\frac{|\mathcal{A}|}{2}$. In all cases, $R_i^T Z_j = 0$. □

Proof of Lemma 3. In this case, Z is allowed to include additional columns which correspond to interaction effects. To form an additional column, one performs element-wise multiplication between columns X_l and Z_j, where where $j = 1, \ldots, \binom{m}{2}$, and $l = 1, \ldots m$. The corresponding column would have entries $Z_{kj} X_{kl}$ for $k = 1, \ldots, N$. As in Lemma 1, it is sufficient to prove that $\sum_{k=1}^{N} R_{ki} Z_{kj} X_{kl} = 0$. Notice that

$$\sum_{k=1}^{N} R_{ki} Z_{kj} X_{kl} = \sum_{k \in \mathcal{K}} Z_{kj} X_{kl} = \sum_{k \in \mathcal{K}} Z_{kj} S_{il} = S_{il} \sum_{k \in \mathcal{K}} Z_{kj} = 0$$

To prove the above, recall that $\mathcal{K}$ is the set of indices k where where row k of X is the same as row i of $\mathbf{S}$. Also, the last equality follows from Lemma 1. □

3.B Appendix: Additional Simulation Results

Table 3.11: Optimal Mixture Proportions, Order, and Response for $m = 4, 5$, $\sigma^2 = 0.05$

m	Mixture	Order	Model	$\mathbf{x}^*$	$\mathbf{a}^*$	y^*
4	1	1	(3)	(0.656,0.344,0,0)	NA	5.264
			(7)	(0.710,0.288,0.001,0.001)	(1,2,3,4)	26.220
		2	(3)	(0.597,0.403,0,0)	NA	5.063
			(7)	(0.209,0.001,0.001,0.789)	(1,2,3,4)	10.293
	2	1	(3)	(0.288,0.201,0.246,0.265)	NA	6.953
			(7)	(0.305,0.214,0.252,0.228)	(1,2,3,4)	27.993
		2	(3)	(0.294,0.195,0.232,0.278)	NA	7.023
			(7)	(0.034,0.083,0.313,0.570)	(1,2,3,4)	14.154
5	1	1	(3)	(0.624,0.376,0,0,0)	NA	4.957
			(7)	(0.787,0.210,0.001,0.001,0.001)	(1,2,3,4,5)	62.750
		2	(3)	(0.639,0.361,0,0,0)	NA	5.013
			(7)	(0.001,0.001,0.001,0.001,0.996)	(1,2,3,4,5)	16.153
	2	1	(3)	(0.234,0.188,0.160,0.190,0.228)	NA	7.500
			(7)	(0.279,0.181,0.149,0.045,0.347)	(1,2,3,4,5)	63.583
		2	(3)	(0.230,0.188,0.152,0.195,0.235)	NA	7.460
			(7)	(0.001.0.001.0.015.0.282.0.701)	(1,2,3,4,5)	19.036

Table 3.12: Optimal Mixture Proportions, Order, and Response for $m = 3, 4$, $\sigma^2 = 1$

m	Mixture	Order	Model	$\mathbf{x}^*$	$\mathbf{a}^*$	y^*
3	1	1	(3)	(0.903,0.039,0.058)	NA	5.198
			(7)	(0.998,0.001,0.001)	(2,1,3)	15.202
		2	(3)	(0.704,0.296,0)	NA	5.918
			(7)	(0.197,0.802,0.001)	(1,3,2)	11.954
	2	1	(3)	(0.336,0.202,0.462)	NA	7.017
			(7)	(0.049,0.273,0.678)	(1,2,3)	14.110
		2	(3)	(0.243,0.380,0.377)	NA	6.019
			(7)	(0.001,0.646,0.353)	(1,2,3)	11.907
4	1	1	(3)	(0.75,0.25,0,0)	NA	5.856
			(7)	(0.967,0.031,0.001,0.001)	(1,2,3,4)	27.388
		2	(3)	(0.475,0.525,0,0)	NA	4.963
			(7)	(0.997,0.001,0.001,0.001)	(2,1,4,3)	15.459
	2	1	(3)	(0.432, 0, 0.237, 0.331)	NA	6.904
			(7)	(0.379,0.286,0.335,0.001)	(1,2,3,4)	28.348
		2	(3)	(0.359,0,0.384,0.256)	NA	7.247
			(7)	(0.156,0.001,0.324,0.519)	(1,2,3,4)	12.224

Table 3.13: Optimal Mixture Proportions, Order, and Response for $m = 5$, $\sigma^2 = 1$

m	Mixture	Order	Model	$\mathbf{x}^*$	$\mathbf{a}^*$	y^*
5	1	1	(3)	(0.724,0,0.228,0,0.048)	NA	4.865
			(7)	(0.996,0.001,0.001,0.001,0.001)	(1,2,3,4,5)	74.475
		2	(3)	(0.686,0.309,0.004,0,0)	NA	4.647
			(7)	(0.001, 0.001, 0.001, 0.001, 0.996)	(1,2,3,4,5)	17.978
	2	1	(3)	(0.227,0.208,0.148,0.216,0.201)	NA	7.770
			(7)	(0.273,0.153,0.001,0.001,0.572)	(1,2,3,4,5)	68.851
		2	(3)	(0.202,0.205,0.121,0.233,0.239)	NA	7.607
			(7)	(0.001,0.001,0.001,0.001,0.996)	(1,2,3,4,5)	22.075

Chapter 4
Constructing Optimal OofA Mixture Designs

4.1 Problem Statement

In general, the OofA Mixutre designs that have been created so far have too many runs. Suppose a mixture design in m components has t mixture experimental runs, where t depends on m. Then, a naive approach would be to try all $t \times m!$ mixture-order combinations, but this scales faster than an exponential rate. Additionally, a mixture need not use all m components, so some of the $t \times m!$ runs may be redundant. For example, Table 4.1 shows the run sizes for OofA Simplex-Lattice Designs with various numbers of components m degrees ℓ. Table 4.1 shows that the run sizes increase rapidly as m and ℓ increase.

Table 4.1: Run Size (N) for Full $\{m, \ell\}$ OofA Simplex Lattice Design

m	l	N
4	3	52
	4	136
6	3	186
	4	816
	5	3006
	6	9276
8	3	456
	4	2864
	5	15688
	6	74208

Given a Mixture-OofA design matrix D_{full} with N rows, we hope to choose a subset

of $n < N$ distinct rows from this design matrix such that the subset of n rows is D-optimal. We are given the target size n, an OofA Mixture model f, and a design matrix $D \in \mathbb{R}^{N \times p + \binom{m}{2}}$, where p is the number of parameters in a mixture model for m components.

Let $\mathcal{D}_n$ be the collection of all matrices that can be made by selecting n distinct (non-repeating) rows from D_{full}. Let $\mathcal{M}_n$ be the collection of all model matrix expansions (using the OofA Mixture model f) of matrices in $\mathcal{D}_n$. Then, as in Section 2.3, we say that M^* is D-optimal if

$$M^* = \underset{M_n \in \mathcal{M}_n}{\arg\min}\ \det((M_n^T M_n)^{-1}) = \underset{M_n \in \mathcal{M}_n}{\arg\max}\ \det(M_n^T M_n)$$

Another criteria of interest in this section is $I-$optimality. As stated in Section 2.3, an $I-$optimal design tries to minimize the average prediction variance over the experimental region. In this case, the experimental region is $\mathcal{S} \times \mathcal{A}$, where $\mathcal{S}$ is some subset of an $(m-1)$ dimensional simplex, and $\mathcal{A}$ is the set of all permutations of $(1, 2, \ldots, m)$. In this case, similar to Goos et al. (2016), M^* is $I-$optimal if

$$M^* = \underset{M_n \in \mathcal{M}_n}{\arg\min} \frac{1}{m! \int_{\mathcal{S}} d\mathbf{x}} \sum_{\mathbf{a} \in \mathcal{A}} \int_{\mathcal{S}} f(\mathbf{x}, \mathbf{a})^T (M_n^T M_n)^{-1} f(\mathbf{x}, \mathbf{a})\ d\mathbf{x} \tag{4.1}$$

The criterion minimized in (4.1) is derived by taking the integral over the space $\mathcal{S} \times A$. In general, for some function $g : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, the volume under g in the region $\mathcal{S} \times \mathcal{A}$ is $\int_{\mathcal{S} \times \mathcal{A}} g(\mathbf{x}, \mathbf{a}) d(\mathbf{x}, \mathbf{a}) = \sum_{\mathbf{a} \in \mathcal{A}} \int_{\mathcal{S}} g(\mathbf{x}, \mathbf{a})\, d\mathbf{x}$, as the set $\mathcal{A}$ is finite and countable. Hence the denominator in (4.1) is derived as $\int_{\mathcal{S} \times \mathcal{A}} = \sum_{\mathbf{a} \in \mathcal{A}} \int_{\mathcal{S}} 1\, d\mathbf{x} = m! \int_{\mathcal{S}} d\mathbf{x}$. The numerator in (4.1) can be written as trace$((M_n^T M_n)^{-1} B)$ where

$$B = \sum_{\mathbf{a} \in \mathcal{A}} \int_{\mathcal{S}} f(\mathbf{x}, \mathbf{a}) f(\mathbf{x}, \mathbf{a})^T\, d\mathbf{x} \tag{4.2}$$

Once an optimal model matrix M^* is found, we know the corresponding optimal design matrix D^*. Throughout this chapter, we will work mainly with the model matrices $M_n \in \mathcal{M}_n$ using a given model f of the general form (3.2) under PWO model coding. Once an optimal design is identified, then the corresponding OofA mixture experiment can be executed and the results used to identify the optimal mixture proportions and order of addition.

4.1.1 Calculation of the $I-$Optimality Criterion

The matrix B in (4.2) needs to be estimated in order to evaluate the $I-$optimality criterion. Unfortunately, this matrix does not have a general closed form in the case where there are constraints placed on the mixture proportions. This happens in the common scenario of single-component constraints $L_i \leq x_i \leq U_i, i = 1, \ldots, m$. In this case, the volume of the region and the integrated volume under $f(\mathbf{x},\, \mathbf{a})$ depends on the constraints. This is further complicated by the inclusion of the orders $\mathbf{a}$, as the choice of OofA Mixture model f and order coding scheme will alter the integral entries of the B matrix.

In general, the entries of B can be estimated using Monte Carlo approximations. For large N, uniformly generate $(\mathbf{x_i}, \mathbf{a_i}) \in \mathcal{S} \times \mathcal{A}, i = 1, 2, \ldots, N$. Then it follows that

$$B \approx \Big(\int_{\mathcal{S}\times\mathcal{A}} d(\mathbf{x}, \mathbf{a}) \Big) \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{x_i}, \mathbf{a_i}) f(\mathbf{x_i}, \mathbf{a_i})^T$$

Note that this estimation of B only requires that we are able to sample points from the experimental region. Therefore, this method can be done in the presence of well-defined constraints, e.g. single-component constraints. If the OofA Mixture model has the general form (3.2) with PWO coding, then it is possible to simplify the calculation of the B matrix. Let $f(\mathbf{x}, \mathbf{a}) = (\eta(\mathbf{x}), g(\mathbf{x}, \mathbf{a}))$. Then

$$\begin{aligned} B &= \int_{\mathcal{S}\times\mathcal{A}} f(\mathbf{x}, \mathbf{a}) f(\mathbf{x}, \mathbf{a})^T \; d(\mathbf{x}, \mathbf{a}) = \int_{\mathcal{S}\times\mathcal{A}} (\eta(\mathbf{x}), g(\mathbf{x}, \mathbf{a}))(\eta(\mathbf{x}), g(\mathbf{x}, \mathbf{a}))^T \; d(\mathbf{x}, \mathbf{a}) \\ &= \int_{\mathcal{S}} \sum_{\mathbf{a}\in\mathcal{A}} \begin{bmatrix} \eta(\mathbf{x})\eta(\mathbf{x})^T & \eta(\mathbf{x}) g(\mathbf{x}, \mathbf{a})^T \\ g(\mathbf{x}, \mathbf{a})\eta(\mathbf{x})^T & g(\mathbf{x}, \mathbf{a}) g(\mathbf{x}, \mathbf{a})^T \end{bmatrix} \; d\mathbf{x} \\ &= \begin{bmatrix} m! \int_{\mathcal{S}} \eta(\mathbf{x})\eta(\mathbf{x})^T \; d\mathbf{x} & 0 \\ 0 & \sum_{\mathbf{a}\in\mathcal{A}} \int_{\mathcal{S}} g(\mathbf{x},\mathbf{a}) g(\mathbf{x},\mathbf{a})^T d\mathbf{x} \end{bmatrix} \end{aligned}$$

Remark 1. The off-diagonal blocks of B are all zero because $\sum_{\mathbf{a}\in\mathcal{A}} g(\mathbf{x}, \mathbf{a}) = 0$ when the order is represented by the PWO variables; as in Lemma 3, this is also true if g contains mixture-order interaction. If the OofA Mixture model does not contain mixture-order interactions, then we may write $g(\mathbf{x},\mathbf{a}) = g(\mathbf{a})$. In this case, the I-optimality simplifies to

$$I(M_n) = \frac{1}{m! \int_{\mathcal{S}} \; d\mathbf{x}} \text{trace} \Big((M_n^T M_n)^{-1} \begin{bmatrix} m! \int_{\mathcal{S}} \eta(\mathbf{x})\eta(\mathbf{x})^T \; d\mathbf{x} & 0 \\ 0 & (\int_{\mathcal{S}} \; d\mathbf{x}) \sum_{\mathbf{a}\in\mathcal{A}} g(\mathbf{a}) g(\mathbf{a})^T \end{bmatrix} \Big)$$

$$= \text{trace}\Bigg((M_n^T M_n)^{-1} \underbrace{\begin{bmatrix} \frac{\int_{\mathcal{S}} \eta(\mathbf{x})\eta(\mathbf{x})^T \, d\mathbf{x}}{\int_{\mathcal{S}} \, d\mathbf{x}} & 0 \\ 0 & \frac{\sum_{\mathbf{a}\in\mathcal{A}} g(\mathbf{a})g(\mathbf{a})^T}{m!} \end{bmatrix}}_{B^*} \Bigg)$$

Remark 2. The upper-left submatrix of B^* is the $B-$matrix used for $I-$optimality of pure mixture experiments. The lower-right submatrix of B is the moment matrix for the full OofA design (Peng et al., 2019), which has a known form of $I_{\binom{m}{2}} + \frac{1}{3}V$, where the $(ij, kl)^{th}$ entry of V for $ij, kl \in \mathcal{P}$ is given by

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

Remark 3. If it is also true that $\mathcal{S}$ is the entire $(m-1)$ dimensional simplex, then the I-optimality criterion reduces to

$$I(M_n) = \text{trace}\Bigg((M_n^T M_n)^{-1} \begin{bmatrix} (m-1)! \int_{\mathcal{S}} \eta(\mathbf{x})\eta(\mathbf{x})^T \, d\mathbf{x} & 0 \\ 0 & \frac{\sum_{\mathbf{a}\in\mathcal{A}} g(\mathbf{a})g(\mathbf{a})^T}{m!} \end{bmatrix} \Bigg) \tag{4.3}$$

where the entries of $\int_{\mathcal{S}} \eta(\mathbf{x})\eta(\mathbf{x})^T \, d\mathbf{x}$ can be computed using the equation

$$\int_{\mathcal{S}} x_1^{p_1} x_2^{p_2} \cdots x_m^{p_m} \, dx_1 dx_2 \ldots dx_m = \frac{\prod_{i=1}^m \Gamma(p_i + 1)}{\Gamma(m + \sum_{i=1}^m p_i)}$$

as indicated in Goos et al. (2016).

4.2 TA Algorithms for OofA Mixture Designs

In this section, we describe how to use Threshold Accpeting (TA) algorithms for selecting ϕ-optimal OofA Mixture designs of a fixed size for some optimaity criterion ϕ. As stated in Winker (2000) and Lyra et al. (2010), TA algorithms initialize a solution and iteratively generate a new solution that is in a "neighborhood" of the current solution. The new solution is then accepted if it strictly improves the objective, or if the decrease in optimality (with respect to the objective function) is within some defined threshold. TA algorithms have very recently seen use with OofA experiments in Winker et al. (2020), so it is natural to consider them in the context of OofA Mixture designs. As a reminder,

pseudo-code for the TA algorithm was reviewed in Chapter 2.

In order to use the TA algorithm in the context of the OofA Mixture problem, it is required to (1) define a neighborhood around a design, i.e., clearly define $\mathcal{N}(D_0)$, and (2) determine how to find the sequence of thresholds $\{t_r : r = 1, 2, \dots\}$. We begin by creating a definition of a neighborhood for an $\{m, \ell\}$ OofA Simplex Lattice Design (OofA SLD).

Definition 1. *Two rows* $\mathbf{d} = [\mathbf{x}, \mathbf{z}(\mathbf{x}, \mathbf{a})]$, $\mathbf{d}' = [\mathbf{x}', \mathbf{z}(\mathbf{x}, \mathbf{a}')]$ *of an OofA SLD in* m *components with degree* ℓ *are* ***pairwise adjacent lattice neighbors*** *if the* ***only*** *change between them is either one of the following:*

1. $||\mathbf{x} - \mathbf{x}'||_2 = \frac{\sqrt{2}}{\ell}$
2. $\mathbf{a}'$ *is obtained by switching one pair of adjacent components in* $\mathbf{a}$.

Definition 2. *Let* D *and* D' *be two* $\{m, \ell\}$ *OofA SLDs with* n *rows. Then* D *and* D' *are* ***pairwise adjacent lattice neighbors*** *if they are identical apart from* k *corresponding pairs of rows* $(\mathbf{d_1}, \mathbf{d'_1}), (\mathbf{d_2}, \mathbf{d'_2}), \dots, (\mathbf{d_k}, \mathbf{d'_k})$*, which are pairwise adjacent lattice neighbors, for some* $k \in \{1, 2, 3, \dots n\}$.

If D and D' are pairwise adjacent lattice neighbors, then we write $D' \in \mathcal{N}_k(D)$. The definition of pairwise adjacent lattice neighbors arises from the fact that the mixture points in an $\{m, \ell\}$ SLD are equally spaced, and the Euclidean distance between any two adjacent mixture points in such a design is $\frac{\sqrt{2}}{\ell}$. Hence, it is natural to conclude that two mixture points $\mathbf{x}, \mathbf{x}'$ are "neighbors" if the distance between them is $\frac{\sqrt{2}}{\ell}$.

The second half of Definition 1 arises by realizing that changing an adjacent pair of components is a relatively small modification to make. In terms of PWO coding, changing the order of adjacent components $j = 1, 2, \dots, m-1$ and $k = j+1$ only requires multiplying z_{jk} by -1. Finally, it should be noted that the definition of pairwise adjacent lattice points implies that $\mathbf{d}$ and $\mathbf{d}'$ are neighbors if they are adjacent in only one of the following regards (and equal in the other): (1) in terms of distance in the simplex or (2) in terms of the pairwise ordering.

It is also possible to define a more general neighborhood, which can be used for the case of constrained mixture proportions.

Definition 3. *Two rows* $\mathbf{d} = [\mathbf{x}, \mathbf{z}(\mathbf{x}, \mathbf{a})]$, $\mathbf{d}' = [\mathbf{x}', \mathbf{z}(\mathbf{x}, \mathbf{a}')]$ *of an OofA Mixture design are* ***pairwise adjacent*** ϵ ***neighbors*** *if the* ***only*** *change between them is one of the following (but not both):*

1. $||\mathbf{x} - \mathbf{x}'||_2 \leq \epsilon$

2. $\mathbf{a}'$ *is obtained by switching one pair of adjacent components in* $\mathbf{a}$.

Definition 3 can be used for any OofA Mixture design, where a value of ϵ needs to be pre-specified. In this definition, $\mathbf{d}$ and $\mathbf{d}'$ if they are nearby in the experimental space of $\mathbf{x}$ (with order constant) or if they are identical in $\mathbf{x}$, but have orderings that differ by a single pairwise switch. One can take ϵ to be a quantile (e.g. $80^{th}, 90^{th}$) of the list of all pairwise distances between all points in the extreme vertices design. If ϵ is too small, then there is a risk of becoming trapped at a local optimum. On the other hand, if ϵ is too large, then the size of the neighborhoods will also increase, possibly causing the algorithm to become inefficient.

Now that neighborhoods are defined, it remains to discuss how the threshold sequence $\{t_r : r = 1, 2, 3, \dots\}$ will be generated. An empirical procedure similar to that of Winker et al. (2020) is used to generate the threshold sequence. The idea behind this algorithm is to find an empirical distribution of the changes in ϕ-efficiency when a design D is replaced with one in $\mathcal{N}_k(D)$. The algorithm first generates a large number of initial designs. Then, for each initial design, find a random neighbor, and store the absolute difference in the ϕ-efficiencies. These differences approximate the distribution of changes in the ϕ-efficiency, and are suitable to use as thresholds. This process is described in Algorithm 5.

Algorithm 5: Algorithm for finding an empirical threshold sequence.

Inputs: number of components m, an OofA Mixture Design D_{full} number of iterations n_i, neighborhood parameter k, and the model $f(\mathbf{x}, \mathbf{a})$

1. Initialize $t = (0, \dots, 0)$ of length $2n_i$.

for $r = 1, 2, \dots, 2n_i$ **do**

2. Form D_0 by choosing n random rows from D_{full}.
3. Find a random $D_1 \in \mathcal{N}_k(D_0)$
4. Let M_0, M_1 be the model matrix expansions of D_0, D_1, respectively.
5. $t_r = |\phi(M_0) - \phi(M_1)|$

end

5. Sort t in descending order.
6. Use the lower 50% of t as the threshold sequence $\{t_r : r = 1, \dots, n_i\}$.

Output The threshold sequence $\{t_r : r = 1, \dots, n_i\}$

Algorithm 5 takes as inputs a full OofA Mixture design, the desired threshold sequence length n_i, the parameter k, which chooses the number of rows exchanged in a neighborhood

$\mathcal{N}_k(D_0)$, and a mixture OofA model $f(\mathbf{x}, \mathbf{a})$. The algorithm first initializes a list t (step 1). Then, the algorithm generates a large number of possible initial designs D_0. For each of these designs, a random neighbor $D_1 \in \mathcal{N}_k(D_0)$ is found, and the absolute difference in ϕ-efficiency is stored in the list (steps 2-5). Finally, these absolute differences are sorted in descending order. Using the entire sequence of absolute differences for the thresholds is inefficient, because the largest values of the sequence are large enough to allow any design to be selected at the beginning. This would mean that many iterations would be wasted on random search steps at the beginning of the algorithm. Instead, the lower 50% of these values are used as the threshold sequence (steps 6-7).

Next, we discuss how to implement the TA Algorithm for the $D-$optimality criterion, denoted $de(M) = |M^T M|$. This is shown in Algorithm 6.

Algorithm 6: TA algorithm for finding D-optimal OofA Mixture design with empirical threshold sequence.

Inputs: number of components m, full OofA Mixture design D_{full}, target size n, number of iterations n_i, and the model $f(\mathbf{x}, \mathbf{a})$

1. Use Algorithm 5 to find the threshold sequence $\{t_r : r = 1, \ldots, n_i\}$
2. Form D_0 by choosing n random rows from D_{full}.
3. Find M_0, the model matrix expansion of D_0.
4. Compute and store $C_0 = (M_0^T M_0)^{-1}$ and $d_0 = |M_0^T M_0|$.

for $r = 1, 2, \ldots, n_{iter}$ **do**

5. Copy $d_1 = d_0, C_1 = C_0$.
6. Randomly choose $D_1 \in \mathcal{N}_k(D_0)$ by sequentially exchanging k random rows of D_0.

for *each exchange of a row* $\mathbf{x}$ *in* D_0 *with some* $\mathbf{y}$ **do**

7. Compute $\Delta(\mathbf{x}, \mathbf{y}) = 1 + v(\mathbf{y}) - v(\mathbf{x}) + v(\mathbf{x}, \mathbf{y})^2 - v(\mathbf{y})v(\mathbf{x})$.
8. $d_1 = d_1 \Delta(\mathbf{x}, \mathbf{y})$.
9. Update $C_1 = C_1 - C_1 F_1 (I_2 + F_2^T C_1 F_1)^{-1} F_2^T C_1$ where $F_1 = [f(\mathbf{y}), -f(\mathbf{x})]$ and $F_2 = [f(\mathbf{y}), f(\mathbf{x})]$.

end

10. Use d_0, d_1 to compute $de(M_0), de(M_1)$.
11. **if** $de(M_1) > de(M_0) - t_r$ **then** Update $D_0 = D_1, M_0 = M_1, d_0 = d_1, C_0 = C_1$.

end

Output D_0 and $de(M_0)$.

Algorithm 6 takes as inputs D_{full}, which is the set of all candidate points. It also takes the desired size n of the final OofA Mixture design, the number of iterations n_i of the algorithm, and the type of model being used $f(\mathbf{x}, \mathbf{a})$. As output, Algorithm 6 returns a design D_0 that is D-efficient, and it also displays the D-efficiency of D_0. Step 1 of Algorithm 6 calls Algorithm 5 to generate the empirical threshold sequence. Step 2 takes a random subset of the rows of D_{full} and uses this as an initial design. Step 3 uses the user-specified model to generate the model matrix M_0 that corresponds to D_0. Once this is done, the matrix inverse C_0 and determinant d_0 are computed in step 4.

In step 5 of Algorithm 6, the values of d_1, C_1 are initialized as copies of d_0, C_0, respectively. In step 6, k randomly selected rows of D_0 are sequentially exchanged to form a neighboring design D_1. Steps 7-9 describe how the determinant d_1 and inverse C_1 are updated for each sequential exchange. Here, Algorithm 6 makes use of some update formulas given in Fedorov (1972) and also given in Meyer and Nachtsheim (1995). Specifically, if a point $\mathbf{x}$ in the model matrix is exchanged with an arbitrary point $\mathbf{y}$ in the design space, then $|M_0^T M_0|$ changes by a multiplicative factor of

$$\Delta(\mathbf{x}, \mathbf{y}) = 1 + v(\mathbf{y}) - v(\mathbf{x}) + v(\mathbf{x}, \mathbf{y})^2 - v(\mathbf{y})v(\mathbf{x}) \tag{4.4}$$

where $v(\mathbf{x}, \mathbf{y}) = f(\mathbf{x})^T (M_0^T M_0)^{-1} f(\mathbf{y})$ and $v(\mathbf{x}) = v(\mathbf{x}, \mathbf{x})$ for ease of notation. Note that $v(\mathbf{x})$ is the prediction variance at the point $\mathbf{x}$. Additionally, the update on line 10 comes from both Fedorov (1972) Meyer and Nachtsheim (1995), who note that after exchanging $\mathbf{x}$ with $\mathbf{y}$, the matrix C_1 can be found without inverting the entire new moment matrix again.

$$C_1 = C_0 - C_0 F_1 (I_2 + F_2^T C_0 F_1)^{-1} F_2^T C_0 \tag{4.5}$$

where $C_1 = (M^T M)^{-1}$ is the inverse of the moment matrix after the exchange, M is the model matrix after the exchange, $C_0 = M_0^T M_0$, and $F_1 = [f(\mathbf{y}), -f(\mathbf{x})], F_2 = [f(\mathbf{y}), f(\mathbf{x})]$. The matrix $(I_2 + F_2^T C_0 F_1)$ is 2×2, so inverting this matrix is computationally easier than finding $(M^T M)^{-1}$ at each iteration of the algorithm. The matrix C_0 is required during each iteration to calculate the prediction variances $v(\mathbf{x}), v(\mathbf{y})$, and $v(\mathbf{x}, \mathbf{y})$, so a quick method of updating this matrix is necessary for this algorithm. Line 11 of the algorithm uses d_0, d_1 to find the D-efficiencies of D_0 and D_1, i.e., the D-efficiencies before and after the k sequential exchanges, respectively. Finally, in step 12, if $de(M_1) > de(M_0) - t_r$ then the exchange is confirmed, otherwise the loop continues.

For Algorithms 5 and 6 to be implemented, a method of quickly finding a random neighbor $D_1 \in \mathcal{N}_k(D_0)$ is required. Since the full design D_{full} is known before the algorithm is executed, it is possible to store a list of all neighbors for each candidate point ahead of time. To generate a random neighbor $D_1 \in \mathcal{N}_k(D_0)$, simply select a row x in D_0, look up the list of neighbors of x, and randomly select one that is not currently in the design. Although it is computationally burdensome to compute the $N \times N$ adjacency matrix, this only needs to be computed once for any full design. Once the adjacency matrix is found, the selection of random neighbors in Step 7 of Algorithm 6 is relatively fast, as the list of neighbors for any row is already stored in the adjacency matrix. We emphasize that our implementation of this procedure enforces the n rows to be unique (no replicates). This restriction may easily be dropped if it is desired.

Algorithm 6 is compared with an updated version of Fedorov's exchange algorithm (Cook and Nachtrheim, 1980) that is outlined in Algorithm 7. Lines 1 to 3 of Algorithm 7 choose a random subset of rows from D_{full} to create an initial design D_0 of size n, find its model matrix expansion M_0, and then compute the initial values of C_0 and M_0. Then for each iteration, each row $\mathbf{x}$ of the current design D_0 is paired with a row $\mathbf{y}$ from D_{full} (that is not currently in D_0) so that $\Delta(\mathbf{x}, \mathbf{y})$ is maximized (Lines 6-8). If $\Delta(\mathbf{x}, \mathbf{y}) > 1$, then the exchange is performed (Lines 9-11). Line 10 keeps track of the multiplicative change in the determinant across the entire iteration. At the conclusion of an iteration, the convergence parameter δ is calculated (Line 12).

Algorithm 7: Fedorov Exchange Algorithm for OofA Mixture Design.

Inputs: number of components m, full OofA Mixture design D_{full} target size n, and the model $f(\mathbf{x}, \mathbf{a})$

1. Form D_0 by choosing n random rows from D_{full}.
2. Find M_0, the model matrix expansion of D_0.
3. Compute and store $C_0 = (M_0^T M_0)^{-1}$.

while $\delta < 10^{-4}$ **do**

4. $d_0 = 1$.

for *each row* $\mathbf{x}$ *of* D_0 **do**

5. Initialize a list Δ.

for *row* $\mathbf{y}$ *in* D_{full} **do**

6. Compute $\Delta(\mathbf{x}, \mathbf{y})^* = 1 + v(\mathbf{y}) - v(\mathbf{x}) + v(\mathbf{x}, \mathbf{y})^2 - v(\mathbf{y})v(\mathbf{x})$.
7. Add $\Delta(\mathbf{x}, \mathbf{y})^*$ to the list Δ.

end

8. $\Delta(\mathbf{x}, \mathbf{y}) = \max_{\mathbf{y} \in D_{full} \setminus D_0}(\Delta)$

if $\Delta(\mathbf{x}, \mathbf{y}) > 1$ **then**

9. $C_0 = C_0 - C_0 F_1 (I_2 + F_2^T C_0 F_1)^{-1} F_2^T C_0$ where $F_1 = [f(\mathbf{y}), -f(\mathbf{x})]$ and $F_2 = [f(\mathbf{y}), f(\mathbf{x})]$.
10. $d_0 = d_0 \Delta(\mathbf{x}, \mathbf{y})$
11. In D_0, exchange row $\mathbf{x}$ for row $\mathbf{y}$.

end

end

12. $\delta = d_0 - 1$

end

Output D_0

Note that if M_t is the model matrix at iteration t, then

$$\delta = \frac{|M_t^T M_t| - |M_{t-1}^T M_{t-1}|}{|M_{t-1}^T M_{t-1}|}.$$

So δ is the proportional increase in the determinant after all of the exchanges in a single iteration are made. It is assumed that the algorithm converges if the proportional increase δ is smaller than a pre-determined threshold. In this implementation, $\delta < 10^{-4}$ was the condition for convergence. As one may imagine, calculating the change in the determinant for all points in the full design D_{full} is computationally burdensome. Also, the Fedorov algorithm runs the risk of becoming trapped at local maximums, because it always performs an exchange if the exchange is beneficial.

Algorithms 6 and 7 can be adjusted to other optimality criteria. Consider linear optimality criteria of the form $\phi(M) = \text{trace}((M^TM)^{-1}B)$. This covers $A-$optimality, where $B = I$, and $I-$optimality, where B is as defined in Equation (4.2). For example, a simple modification of Algorithm 6 is to delete lines 7, 8 and simply update the inverse $C_0 = (M_0^TM_0)^{-1}$ by equation (4.5), and then use this inverse to find $\phi(M_0), \phi(M_1)$ in line 10. Another option is to use a fast-updating formula. Let $\Delta_\phi(\mathbf{x}, \mathbf{y})$ be the decrease in $\phi(M)$ that results from exchanging row $\mathbf{x}$ with row $\mathbf{y}$. Meyer and Nachtsheim (1995) provide a closed form for $\Delta_\phi(\mathbf{x}, \mathbf{y})$. Let

$$\Delta_\phi(\mathbf{x}, \mathbf{y}) = \frac{[1 - v(\mathbf{x})]L(\mathbf{y}) + v(\mathbf{x}, \mathbf{y})[L(\mathbf{y}, \mathbf{x}) + L(\mathbf{x}, \mathbf{y})] - [1 - v(\mathbf{x})]L(\mathbf{x})}{\Delta(\mathbf{x}, \mathbf{y})}$$

where $\Delta(\mathbf{x}, \mathbf{y})$ is the change in $D-$optimality from Equation (4.4). The function L is given by

$$L(\mathbf{x}, \mathbf{y}) = \text{trace}((C_0 f(\mathbf{x}) f(\mathbf{y})^T C_0)B)$$

and $L(\mathbf{x}) = L(\mathbf{x}, \mathbf{x})$ for ease of notation. So, to find $\phi-$optimal designs, Algorithms 6 and 7 would need to add the extra step of computing $\Delta_\phi(\mathbf{x}, \mathbf{y})$ after computing $\Delta(\mathbf{x}, \mathbf{y})$. This allows for fast updating of a linear optimality criterion without inverting the matrix $M_0^TM_0$ again after each exchange. With this change, one can search for designs that minimize the $I-$ or $A-$optimality criterion.

4.3 Impact of Algorithm Parameters

In this section, we focus on the implementation of Algorithm 6 for the OofA SLD. In this case, one needs to set the number of iterations and the degree of the OofA SLD. The degree may be limited by real world constraints, but the number of iterations is only limited by computing time. In this section, we examine (1) the convergence of the TA algorithm as the number of iterations increases, (2) how the performance of the TA algorithm changes as n increases relative to N, and (3) the efficiency of small double-point designs for various m and ℓ. A double-point design has $n = 1 + 2p$ runs, where p is the number of model parameters. We measure the performance of the TA algorithm by using the relative D-efficiency of the n-run design to that of the full OofA Mixture design. The $D-$efficiency criterion is used for speed of computation and convenience; these calculations can be performed for any optimality criterion ϕ that may be quickly

updated. In general, we observed that as the number of exchanges k increased, the relative D-efficiency tended to decrease. For this reason, we kept the number of exchanges at $k = 1$. For simplicity, we use the additive form of the OofA Mixture model (3.2).

We first examine the convergence of the TA algorithm as the number of iterations increases. In Table 4.2, we examine the interquartile range (IQR) and the standard deviation (in parentheses) of the distribution of relative D-efficiencies for 100 different seeds for all combinations of m, ℓ where $\ell \leq \min(m, 5)$. In this table, $n = 1 + 2p$, where $p = m + 2\binom{m}{2}$ is the number of parameters in the Model (3.2). In all but two cases (noted below), the value of n did not greatly effect convergence of the algorithm.

Table 4.2: IQR (and Standard Deviation) for Relative D-Efficiencies of Designs Chosen by TA Algorithm

		Number of Iterations		
m	l	5000	20000	100000
4	3	0.1070 (0.0836)	0.1065 (0.0859)	0.1151 (0.0901)
	4	0.6749 (0.3632)	0.0003 (0.2699)	<0.0001 (0.2760)
6	3	0.1777 (0.1299)	0.1440 (0.1261)	0.1259 (0.1151)
	4	2.0549 (1.5209)	0.6019 (0.4242)	0.1725 (0.1538)
	5	3.3511 (2.5339)	3.6768 (2.3487)	0.3751 (0.3702)
8	3	0.1432 (0.1077)	0.1432 (0.1078)	0.1663 (0.1480)
	4	2.2928 (1.5576)	1.1680 (0.9933)	0.3294 (0.2812)
	5	2.2636 (1.8204)	3.2879 (2.1425)	1.9445 (1.3119)

In Table 4.2, in most cases, the IQR and standard deviation of the relative D-efficiencies decrease as the number of iterations increases. 100000 iterations appears sufficiently large enough to reduce the spread (measured by IQR or standard deviation) of the distribution of relative D-efficiencies. The main exceptions to this case are when $m = 4, 8$ and $\ell = 3$, where the standard deviation increases from 20000 to 100000 iterations. In these cases, increasing the target run size n improves the convergence of the algorithm. For instance, when $m = 8, \ell = 3$, if n is increased to $1 + 3p$ (42% of N), then the standard deviations become 0.0499 (5000 iterations), 0.033 (20000 iterations), and 0.033 (100000 iterations). See Table B5 in Appendix 4.B for more summary statistics on the distribution of relative $D-$efficiencies.

Figure 4.1 shows the distribution of relative D-efficiencies (to the full design) for designs selected by the TA algorithm for 100 different random seeds with 5000 , 20000 , and 100000 iterations. In most cases, as the number of iterations increases, the spread of the distribution decreases, indicating convergence. Also, as the number of iterations

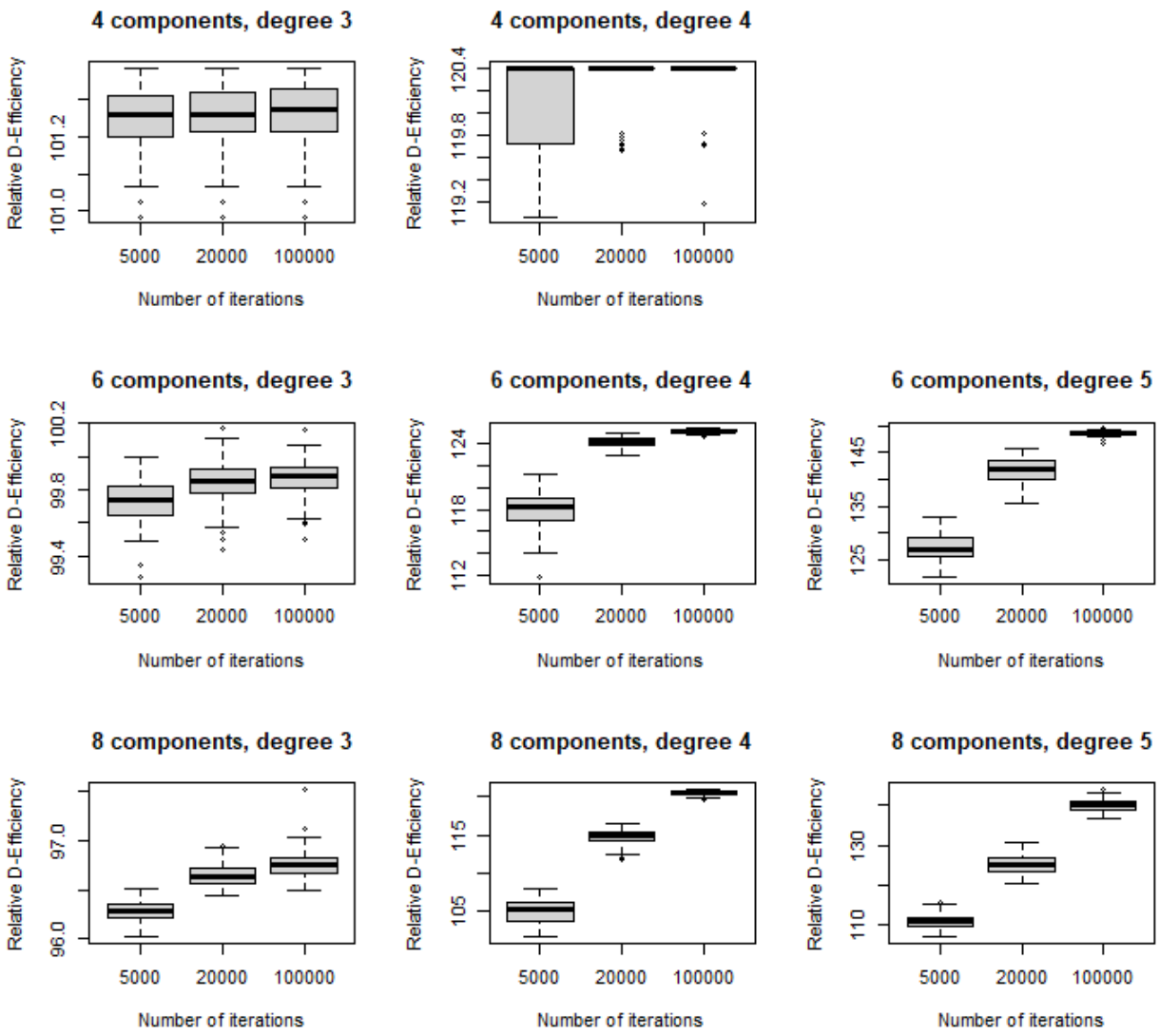
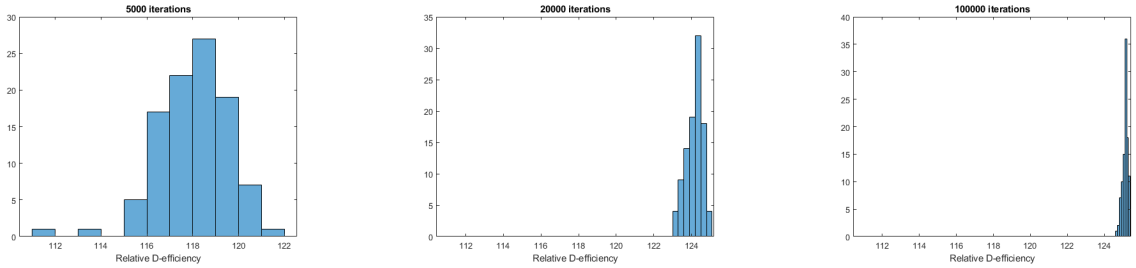


Figure 4.1: Relative D-efficiency Comparison for Various $\{m, \ell\}$ OofA SLD Designs at Varying Number of Iterations of the TA Algorithm.

increases, the center of the distribution typically increases. An example of this is more clearly illustrated by the histograms in Figure 4.2 for $m = 6, \ell = 4$. Here, the distribution of relative D-efficiencies (to the full design) for designs selected by the TA algorithm for 100 different random seeds with 5000 (panel a), 20000 (panel b), and 100000 (panel c) iterations, respectively. As the number of iterations increases, the spread of the distribution decreases, indicating convergence. Also, as the number of iterations increases, the center of the distribution shifts to the right.

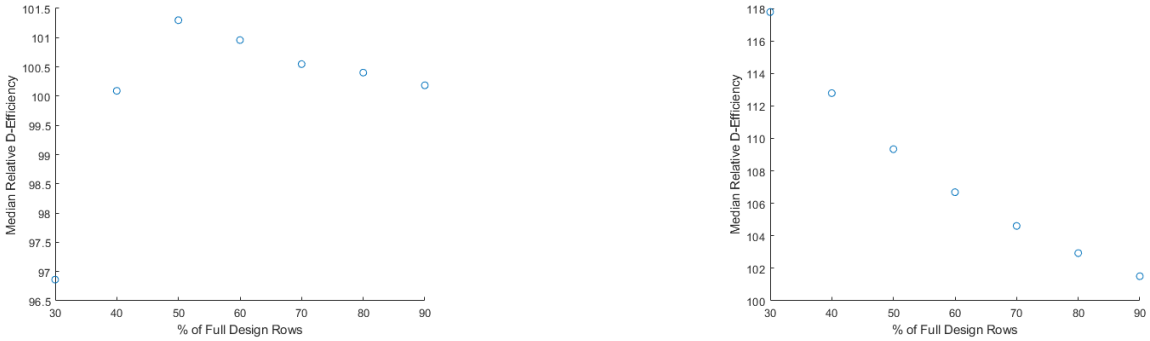
We also studied how the relative D-efficiency changes as the target run size n increases



(a) 5000 iterations. (b) 20000 iterations. (c) 100000 iterations.

Figure 4.2: Convergence of TA Algorithm, $m = 6, \ell = 4, n = 1 + 2p$

(as a percentage of N). We considered all combinations of m, ℓ with $l \leq m$ for $m = 4, 6, 8$. For each m, ℓ, we ran the TA algorithm for $n = 0.3N, 0.4N, 0.5N, 0.6N, 0.7N, 0.8N$, and $0.9N$ for 10 random seeds. For each case, the TA algorithm used 100000 iterations. In each case, we found the median relative D-efficiency and examined a plot of the results. Figure 4.3 shows these plots for $m = 6, \ell = 3$ (panel a) and $m = 6, \ell = 4$ (panel b).



(a) $l = 3$ (b) $l = 4$

Figure 4.3: Median Relative D-efficiency of TA Algorithm, $m = 6, l = 3, 4$

The x-axes in Figure 4.3 are $100(n/N)\%$. In panel (a), we see that the relative D-efficiency increases to a peak at $n = 0.5N$, and then decreases. In panel (b), the highest relative D-efficiency occurs at $n = 0.3N$, and then the relative D-efficiency quadratically decreases (to around 100%) as n increases.

In Table 4.3, we see that the best median relative D-efficiencies are achieved when l is highest, regardless of the target run size n. When the degree of the design is 3, it appears that the best relative D-efficiencies are achieved when $n = 0.5N$, i.e., half the size of the full design. In the other cases, we observed that the median relative D-efficiency decreases as n increases. In these cases, a smaller design is preferred.

Table 4.3: Median Relative D-efficiencies for TA Algorithm

		n						
m	l	0.3N	0.4N	0.5N	0.6N	0.7N	0.8N	0.9N
4	3	*	98.96	101.67	101.40	100.79	100.28	99.94
	4	118.76	115.90	114.03	110.65	107.47	104.79	102.10
6	3	96.86	100.09	101.30	100.96	100.55	100.40	100.19
	4	117.83	112.85	109.38	106.71	104.63	102.95	101.52
	5	122.90	119.31	115.94	111.40	107.70	104.69	102.19
8	3	97.41	99.69	100.93	100.90	100.71	100.47	100.21
	4	111.39	108.40	106.32	104.80	103.62	102.50	101.25
	5	118.56	113.45	109.88	107.25	105.25	103.57	101.84

* This case cannot be examined because $n < p + 1$.

We are also interested in seeing how well a double point design (where $n = 1 + 2p$) performs relative to the full design. To investigate this, we fix the number of components m. For each m, we set the target run size $n = 1 + 2p$, and compare distributions of the TA designs made with different degrees. The cases we examined are shown in Table 4.4.

Table 4.4: Median Relative D-efficiency Comparison for Varying SLD Degrees (100000 iterations)

m	l	n	100(n/N)%	Q_{50}
4	3	33	63.46	101.27
	4	33	24.26	120.39
6	3	73	39.24	99.88
	4	73	8.95	125.31
	5	73	2.43	148.71
8	3	129	28.29	96.83
	4	129	4.50	120.52
	5	129	0.82	140.10

Table 4.4 shows the median relative D-efficiencies of 100 designs generated from the TA algorithm. In each row, m, ℓ, and n were fixed. These results suggest that when $n = 1 + 2p$, using a design with higher degree (larger ℓ) will result in a higher relative D-efficiency to the full design.

4.4 Comparison with Exchange Algorithm

It is of interest to compare the TA algorithm (Algorithm 6) to an exchange algorithm, as both of these can try to find an optimal subset of candidate points from full OofA Mixture Designs. Exchange algorithms are widely used to find experimental designs for a fixed number of runs under an optimality criterion. Exchange algorithms are appealing, as many optimality criteria have fast update formulas when a single row of the design matrix is exchanged. For suitable candidate points, exchange algorithms converge to a design with high optimality. One of the most popular exchange algorithms is the modified Fedorov algorithm (Algorithm 7). Algorithms 6 and 7 were implemented in Matlab. The TA algorithm used $k = 1$, i.e., only one row was exchanged at a time. To compare the D-efficiency, we rely on the relative D-efficiency of a design D to the full OofA Mixture design D_{full}. Let M and M_{full} be the model matrices of D, D_{full}, respectively. Then the relative D-efficiency of D to D_{full} is defined as

$$\text{Relative D-efficiency} = 100\%\Big(\frac{\frac{1}{n}|M^TM|^{1/p}}{\frac{1}{N}|M_{full}^TM_{full}|^{1/p}}\Big)$$

To compare $I-$efficiencies, we use the relative $I-$efficiency, which is defined as

$$\text{Relative I-efficiency} = 100\%\Big(\frac{\text{trace}((M_{full}^TM_{full}/N)^{-1}B)}{\text{trace}((M^TM/n)^{-1}B)}\Big)$$

The relative $I-$efficiency is written so that values above 100% imply that the design D has lower integrated prediction variance than the full design, and is therefore more optimal. Similarly, values below 100% imply that the full design is more optimal than the n-run design D. In Section 4.4.1, we show an example comparing these algorithms for a simplex-lattice design. In Section 4.4.2, these algorithms are compared in an example with single-component constraints and an extreme vertices design.

4.4.1 OofA Simplex-Lattice Design

Both algorithms were run for $m = 4$ components, degree $\ell = 3$, and target sample size $n = 30$ ($N = 52$) for the additive OofA Mixture model (3.2). The Fedorov algorithm converged in 3 iterations. The TA algorithm was run for 100,000 iterations. The design chosen by the Fedorov algorithm 7 had a relative D-efficiency of 101.6243% to the full design. The same initial points were used for each algorithm. Since the TA algorithm 6

is stochastic, it was executed for 1000 different random seeds. The empirical distribution of the relative D-efficiencies of these 1000 designs is shown in Figure 4.4. Quantiles of the 1000 relative D-efficiencies are shown in Table 4.5.

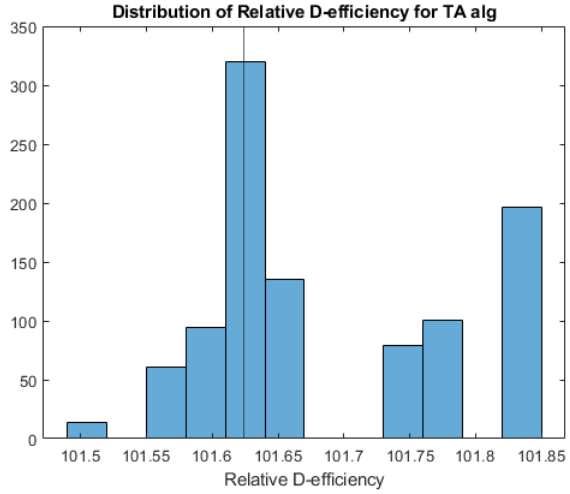


Figure 4.4: Relative D-efficiencies of TA Designs for 1000 random seeds. The vertical line represents the relative efficiency of the design obtained by the Fedorov algorithm.

Table 4.5: Quantiles of the relative D-efficiencies of 1000 designs generated by Algorithm 6

$Q_{0.10}$	$Q_{0.25}$	Median	$Q_{0.75}$	$Q_{0.90}$
101.59%	101.62%	101.65%	101.76%	101.85%

In this case, the empirical distribution appears to be left-skewed. Table 4.5 shows that well over 50% of the designs found using the TA algorithm resulted in a relative D-efficiency at least as good as the Fedorov algorithm. In fact, 71% of the designs generated by the TA algorithm had relative D-efficiencies greater than or equal to the Fedorov algorithm. The IQR is $Q_{0.75} - Q_{0.25} = 0.1432\%$, which indicates that the variability in the relative D-efficiency that occurs due to the change in the seed appears to be small (less than 1%). In this case, it is clear that the TA algorithm is likely to provide a solution that is at least marginally better than that of the Fedorov algorithm. The best designs found by the TA and Fedorov algorithms are provided in Appendix 4.B.

In this example, we see that the relative D-efficiencies are above 100%. In this specific case, this suggests that under the additive OofA Mixture model and the D-criterion, it may be more efficient to use fewer runs than the full design. In Section 5.2, we will see this happen again in many cases.

We also investigated the relative $I-$efficiencies in this scenario. Since the experimental region covers the entire simplex and the additive OofA Mixture model was used, the closed form of the $I-$optimality criterion Equation (4.3) can be used. For comparison, we also used the Monte Carlo approximation of the $I-$optimality. The TA algorithm was executed for the same 1000 random seeds as before, and it was compared to the Federov algorithm on the same set of initial design points.

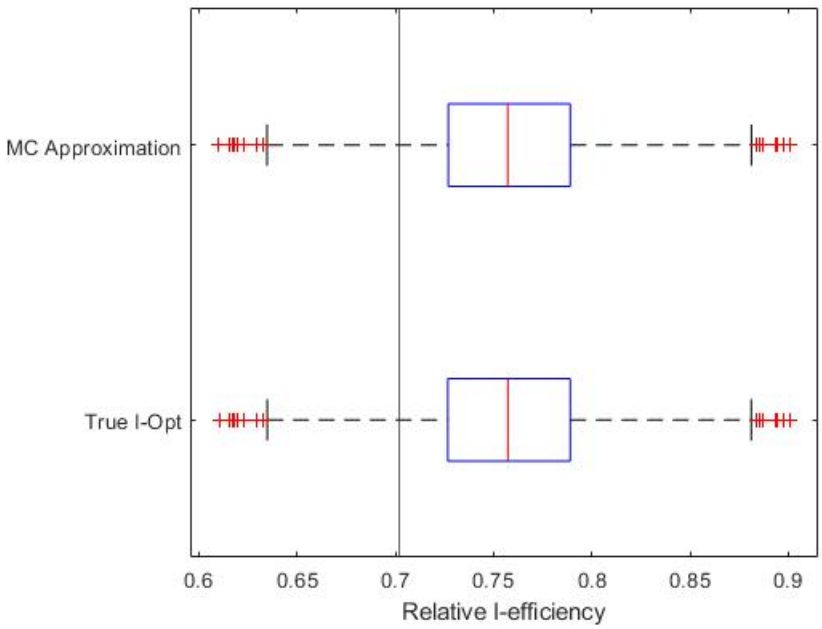


Figure 4.5: Relative I-efficiencies of TA Designs for 1000 random seeds. The vertical line represents the relative efficiency of the design obtained by the Fedorov algorithm.

Figure 4.5 shows the distribution of the relative $I-$efficiencies found by the TA algorithm. The upper boxplot shows these efficiencies under the Monte Carlo approximation of the $B-$matrix, where the lower boxplot shows the efficiencies under the true $B-$matrix. There is little difference between these two plots, which suggests that the Monte Carlo approximation is doing well in this case. The vertical line represents the relative $I-$efficiency of the Federov design, which is about 70%. It is clear that over 75% of the designs identified by the TA algorithm had higher relative $I-$efficiency than that of the Federov design.

4.4.2 OofA Extreme Vertices Design

We borrow an example of an extreme vertices design from (Cornell, 1990), where we have $m = 4$ components with single component constraints $0.4 \leq x_1 \leq 0.8, 0.1 \leq x_2 \leq 0.5, 0.05 \leq x_3 \leq 0.3, 0.05 \leq x_4 \leq 0.3$. The polyhedron resulting from these constraints is illustrated in Figure 4.6.

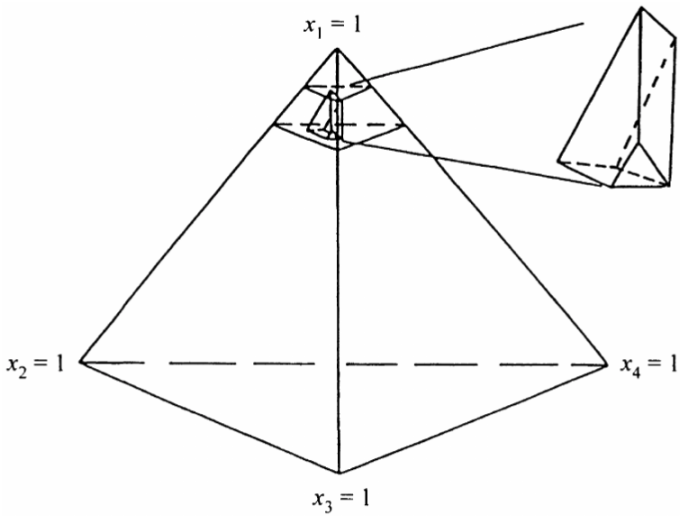


Figure 4.6: Polyhedron Defined by $0.8 \leq x_1 \leq 0.9, 0.05 \leq x_2 \leq 0.15, 0.02 \leq x_3 \leq 0.10, 0.03 \leq x_4 \leq 0.05$, from Cornell (1990)

As shown in Figure 4.6, the region defined by these constraints is not a simplex. The EVD used by Cornell (1990) consists of 15 runs, 8 of which are vertex points on the polyhedron, 6 of which are face centroids, and one overall centroid. See Appendix 4.A for more details on this design. Since all of these points lie in the interior of the simplex, the resulting OofA Extreme Vertices Design (OofA EVD) has $N = 15m! = 360$ runs. We try to find $D-$optimal and $I-$optimal designs with $n = N/6 = 60$ runs for Model (3.5), which includes both main effects and mixture-order interactions. To evaluate the $I-$optimality criterion, we used a Monte Carlo approximation of the $B-$matrix by simulating $100,000$ points in the constrained region defined in Figure 4.6. As in Section 5.3.1, we use the same initial points for the Fedorov and TA algorithms, and the TA algorithm was run for 100,000 iterations and 1000 different random seeds. For this implementation, ϵ was chosen to be the 90^{th} percentile of all mixture points from the extreme vertices design. The Fedorov algorithm selected a design with a relative D-efficiency of 120.5293%.

Based on Table 4.6, we can see that over 90% of the designs generated by the TA

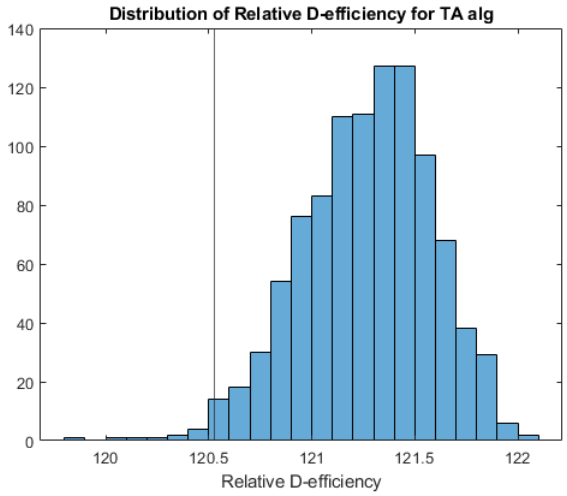


Figure 4.7: Relative D-efficiencies of TA OofA EVDs for 1000 random seeds. The vertical line represents the relative efficiency of the design obtained by the Fedorov algorithm.

Table 4.6: Quantiles of the relative D-efficiencies of 1000 OofA EVDs generated by Algorithm 6

$Q_{0.10}$	$Q_{0.25}$	Median	$Q_{0.75}$	$Q_{0.90}$
120.86%	121.07%	121.29%	121.49%	121.66%

algorithm (roughly 98%) had higher relative D-efficiency than the design generated by the Fedorov algorithm. It is clear that in this case, the TA algorithm is preferred. The best designs found by the TA and Fedorov algorithms are provided in Appendix 4.B.

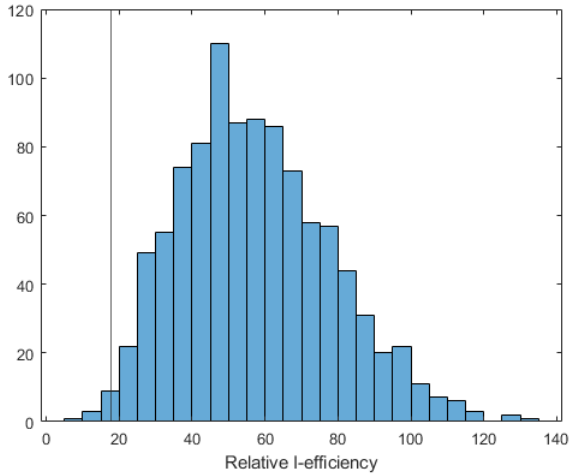


Figure 4.8: Relative I-efficiencies of TA OofA EVDs for 1000 random seeds. The vertical line represents the relative efficiency of the design obtained by the Fedorov algorithm.

Figures 4.7 and 4.8 show the distribution of the relative $D-$ and $I-$efficiencies for the TA designs. In both figures, the relative efficiency of the Federov design is represented by a vertical line. In both figures, the majority of designs found by the TA algorithm have higher relative efficiency than the Federov design. In this case, the distribution for the relative $I-$efficiencies is more spread out, though we still identify several designs with relative $I-$efficiency above 100%. This example also illustrates that (1) we can search for $D-$optimal or $I-$optimal OofA Mixture designs when there are constraints on the mixture proportions and (2) our methods also work on models that include mixture-order interaction terms.

4.4.3 Summary

In this section, we see that the TA algorithm is a useful and flexible method for searching for D-optimal OofA Mixture designs. In both examples, the majority of designs found by the TA algorithm had higher D-efficiencies than those found by the Federov algorithm. The first example showed a classical problem where the design space of the mixture proportions was the $(m-1)$-dimensional simplex, while the second example dealt with the case where the design space was a polyhedron within the same simplex. Additionally, these examples show that our methods can be applied to an OofA mixture model with or without interaction terms. We also see that in both of these examples, reducing the number of runs improves the D-efficiency relative to the full design. This intuitively suggests that the full design has too many replicates, and not all of these runs are needed.

4.5 Conclusion

To the best of our knowledge, this is the first attempt at finding small-run OofA Mixture experiments according to an optimality criterion. Moreover, the methodology proposed here allows for flexibility in the choice of the target run size n. This approach can also be used to create an OofA Mixture design from a variety of existing mixture designs, even in the case of single component constraints. Furthermore, the methods in this chapter are not necessarily limited to the D-optimality criterion; as long as it is possible to find the change in optimality at each iteration of the TA algorithm, then these methods may be applied. This research also provides a useful tool for experimental design in the fields of biochemistry, chemical engineering, and food science, where the response may depend on both the mixture proportions and their addition order.

In this chapter, the TA algorithm was used to search for OofA Mixture designs of a fixed run size according to the $D-$optimality and $I-$optimality criteria. In Sections 5.2 and 5.3, we found that in many cases, using a design with a small fraction of the $t \times m!$ runs (for t mixture points) results in much higher D-efficiency relative to the full OofA Mixture design. It is intuitive that reducing the run size will lead to better designs relative to the full design, as the full OofA Mixture design has many replicates. In Section 5.3, we gave examples where the TA algorithm outperforms the classical Fedorov approach in terms of selecting a design with higher relative D-efficiency.

The methods proposed in this chapter can be used in many practical situations. The general definition of a neighborhood proposed in Section 5.1.3 is applicable to any OofA Mixture design for a suitably chosen radius $\epsilon > 0$. Moreover, as long as a set of feasible mixture points for an experiment can be listed, the TA algorithm proposed in this chapter can be used to search for $D-$ or $I-$optimal designs. This makes the method appealing for cases with single-component constraints, as shown in Section 5.3.2. Additionally, the TA algorithm can be applied to more design-specific neighborhoods, such as the simplex-lattice neighborhood constructed in this chapter.

For implementing this algorithm, it is recommended to use a target run size of at least $n = 1 + 2p$, where p is the number of model parameters. In the case of the simplex-lattice design, as the degree l increases, the TA algorithm becomes more computationally efficient when operating on a larger run size, making it easier to achieve convergence in fewer iterations. It is debatable if this is a practical concern, as lattice designs with high degrees are not often used. Also, in Section 5.2, it appears that lower run sizes were shown to have higher relative D-efficiency to the full design, except in the case where $l = 3$, where a half-fraction design seems to be optimal.

4.A Appendix: Extreme Vertices Design from Cornell (1990)

In one of Cornell's experiments, a tropical drink was made by mixing watermelon juice (x_1), orange juice (x_2) pineapple juice (x_3), and grapefruit juice (x_4). This design was made for points under the constraints $0.4 \leq x_1 \leq 0.8, 0.1 \leq x_2 \leq 0.5, 0.05 \leq x_3 \leq 0.3, 0.05 \leq x_4 \leq 0.3$ with the goal of maximizing the response, which was the average flavor score of the beverage. Table A1 provides the design from Cornell (1990) that is used in Section 5.3.2 as a basis for constructing an OofA Mixture design.

Table A1: Extreme Vertices Design Data from Cornell (1990)

x_1	x_2	x_3	x_4
0.40	0.10	0.30	0.20
0.80	0.10	0.05	0.05
0.40	0.50	0.05	0.05
0.40	0.25	0.05	0.30
0.40	0.25	0.30	0.05
0.40	0.10	0.20	0.30
0.55	0.10	0.05	0.30
0.55	0.10	0.30	0.05
0.58	0.24	0.05	0.13
0.54	0.24	0.17	0.05
0.40	0.24	0.18	0.18
0.54	0.10	0.18	0.18
0.45	0.15	0.10	0.30
0.45	0.15	0.30	0.10
0.49	0.19	0.16	0.16

4.B Appendix: More Results for Sections 3 and 4

Table B1: $D-$Optimal OofA Mixture Design Selected by Federov Algorithm, Example 4.1

x_1	x_2	x_3	x_4	z_{12}	z_{13}	z_{14}	z_{23}	z_{24}	z_{34}
1.00	0.00	0.00	0.00	0	0	0	0	0	0
0.00	1.00	0.00	0.00	0	0	0	0	0	0
0.00	0.00	1.00	0.00	0	0	0	0	0	0
0.00	0.00	0.00	1.00	0	0	0	0	0	0
0.67	0.33	0.00	0.00	1	0	0	0	0	0
0.33	0.67	0.00	0.00	-1	0	0	0	0	0
0.67	0.00	0.33	0.00	0	1	0	0	0	0
0.00	0.67	0.33	0.00	0	0	0	1	0	0
0.33	0.00	0.67	0.00	0	-1	0	0	0	0
0.00	0.33	0.67	0.00	0	0	0	-1	0	0
0.67	0.00	0.00	0.33	0	0	-1	0	0	0
0.00	0.67	0.00	0.33	0	0	0	0	1	0
0.00	0.00	0.67	0.33	0	0	0	0	0	1
0.33	0.00	0.00	0.67	0	0	1	0	0	0
0.00	0.33	0.00	0.67	0	0	0	0	-1	0
0.00	0.00	0.33	0.67	0	0	0	0	0	-1
0.33	0.33	0.33	0.00	-1	-1	0	-1	0	0
0.33	0.33	0.33	0.00	-1	1	0	1	0	0
0.33	0.33	0.33	0.00	1	1	0	-1	0	0
0.33	0.33	0.33	0.00	1	1	0	1	0	0
0.33	0.33	0.00	0.33	-1	0	-1	0	-1	0
0.33	0.33	0.00	0.33	-1	0	1	0	1	0
0.33	0.33	0.00	0.33	1	0	1	0	-1	0
0.33	0.00	0.33	0.33	0	1	-1	0	0	-1
0.33	0.00	0.33	0.33	0	-1	-1	0	0	1
0.33	0.00	0.33	0.33	0	-1	1	0	0	1
0.33	0.00	0.33	0.33	0	1	1	0	0	1
0.00	0.33	0.33	0.33	0	0	0	-1	-1	-1
0.00	0.33	0.33	0.33	0	0	0	-1	1	1
0.00	0.33	0.33	0.33	0	0	0	1	1	-1

Table B2: Best $D-$Optimal OofA Mixture Design Found by TA Algorithm, Example 4.1

x_1	x_2	x_3	x_4	z_{12}	z_{13}	z_{14}	z_{23}	z_{24}	z_{34}
1.00	0.00	0.00	0.00	0	0	0	0	0	0
0.00	1.00	0.00	0.00	0	0	0	0	0	0
0.00	0.00	1.00	0.00	0	0	0	0	0	0
0.00	0.00	0.00	1.00	0	0	0	0	0	0
0.67	0.33	0.00	0.00	1	0	0	0	0	0
0.33	0.67	0.00	0.00	-1	0	0	0	0	0
0.67	0.00	0.33	0.00	0	1	0	0	0	0
0.00	0.67	0.33	0.00	0	0	0	1	0	0
0.33	0.00	0.67	0.00	0	-1	0	0	0	0
0.00	0.33	0.67	0.00	0	0	0	-1	0	0
0.67	0.00	0.00	0.33	0	0	1	0	0	0
0.00	0.67	0.00	0.33	0	0	0	0	1	0
0.00	0.00	0.67	0.33	0	0	0	0	0	1
0.33	0.00	0.00	0.67	0	0	-1	0	0	0
0.00	0.33	0.00	0.67	0	0	0	0	-1	0
0.00	0.00	0.33	0.67	0	0	0	0	0	-1
0.33	0.33	0.33	0.00	-1	-1	0	-1	0	0
0.33	0.33	0.33	0.00	-1	-1	0	1	0	0
0.33	0.33	0.33	0.00	-1	1	0	1	0	0
0.33	0.33	0.33	0.00	1	1	0	-1	0	0
0.33	0.33	0.00	0.33	1	0	-1	0	-1	0
0.33	0.33	0.00	0.33	-1	0	-1	0	1	0
0.33	0.33	0.00	0.33	1	0	1	0	-1	0
0.33	0.33	0.00	0.33	1	0	1	0	1	0
0.33	0.00	0.33	0.33	0	1	-1	0	0	-1
0.33	0.00	0.33	0.33	0	-1	-1	0	0	1
0.33	0.00	0.33	0.33	0	1	1	0	0	1
0.00	0.33	0.33	0.33	0	0	0	-1	-1	-1
0.00	0.33	0.33	0.33	0	0	0	-1	1	1
0.00	0.33	0.33	0.33	0	0	0	1	1	-1

Table B3: $D-$Optimal OofA Mixture Design Selected by Federov Algorithm, Example 4.2

x_1	x_2	x_3	x_4	z_{12}	z_{13}	z_{14}	z_{23}	z_{24}	z_{34}
0.40	0.10	0.30	0.20	-1	-1	-1	-1	-1	-1
0.40	0.10	0.30	0.20	1	1	-1	-1	-1	-1
0.40	0.10	0.30	0.20	-1	-1	1	1	1	1
0.40	0.10	0.30	0.20	1	1	1	1	1	1
0.80	0.10	0.05	0.05	1	-1	-1	-1	-1	-1
0.80	0.10	0.05	0.05	-1	1	-1	1	-1	-1
0.80	0.10	0.05	0.05	1	1	-1	-1	-1	-1
0.80	0.10	0.05	0.05	-1	-1	1	-1	1	1
0.80	0.10	0.05	0.05	1	-1	1	-1	-1	1
0.80	0.10	0.05	0.05	-1	-1	-1	1	1	-1
0.80	0.10	0.05	0.05	-1	-1	-1	1	1	1
0.80	0.10	0.05	0.05	-1	1	1	1	1	-1
0.80	0.10	0.05	0.05	1	1	1	1	-1	-1
0.80	0.10	0.05	0.05	1	1	1	-1	1	1
0.80	0.10	0.05	0.05	1	1	1	1	1	1
0.40	0.50	0.05	0.05	1	-1	-1	-1	-1	-1
0.40	0.50	0.05	0.05	-1	-1	-1	1	-1	-1
0.40	0.50	0.05	0.05	-1	1	-1	1	-1	-1
0.40	0.50	0.05	0.05	1	-1	-1	-1	-1	1
0.40	0.50	0.05	0.05	-1	-1	-1	-1	1	1
0.40	0.50	0.05	0.05	-1	-1	1	-1	1	1
0.40	0.50	0.05	0.05	-1	1	-1	1	1	-1
0.40	0.50	0.05	0.05	-1	-1	1	1	1	1
0.40	0.50	0.05	0.05	1	1	1	-1	-1	-1
0.40	0.50	0.05	0.05	1	1	1	-1	-1	1
0.40	0.50	0.05	0.05	1	1	1	1	1	-1
0.40	0.50	0.05	0.05	1	1	1	1	1	1
0.40	0.25	0.05	0.3	-1	-1	-1	1	-1	-1
0.40	0.25	0.05	0.3	-1	-1	-1	-1	1	1
0.40	0.25	0.05	0.3	1	1	1	-1	-1	1
0.40	0.25	0.05	0.3	1	1	1	1	1	-1
0.40	0.25	0.30	0.05	-1	-1	-1	1	-1	-1
0.40	0.25	0.30	0.05	1	1	-1	1	-1	-1
0.40	0.25	0.30	0.05	-1	-1	-1	-1	1	1
0.40	0.25	0.30	0.05	-1	1	1	1	1	1
0.40	0.25	0.30	0.05	1	1	1	-1	-1	-1
0.40	0.10	0.20	0.3	-1	1	-1	1	-1	-1
0.40	0.10	0.20	0.30	1	1	-1	1	-1	-1
0.40	0.10	0.20	0.30	-1	-1	-1	-1	1	1
0.40	0.10	0.20	0.30	1	-1	1	-1	-1	1
0.40	0.10	0.20	0.30	-1	1	1	1	1	1
0.40	0.10	0.20	0.30	1	1	1	1	1	-1
0.55	0.10	0.05	0.30	-1	-1	-1	-1	-1	-1
0.55	0.10	0.05	0.30	1	-1	-1	-1	-1	1
0.55	0.10	0.05	0.30	-1	1	-1	1	1	-1
0.55	0.10	0.05	0.30	-1	-1	1	1	1	1
0.55	0.10	0.05	0.30	1	1	1	-1	-1	-1
0.55	0.10	0.05	0.30	1	1	1	1	1	1
0.55	0.10	0.30	0.05	-1	1	-1	1	-1	-1
0.55	0.10	0.30	0.05	1	-1	1	-1	-1	1
0.55	0.10	0.30	0.05	-1	-1	-1	1	1	-1
0.55	0.10	0.30	0.05	1	1	1	-1	1	1
0.58	0.24	0.05	0.13	-1	-1	-1	-1	-1	1
0.58	0.24	0.05	0.13	1	1	1	1	1	-1
0.54	0.24	0.17	0.05	-1	-1	-1	-1	-1	1
0.54	0.24	0.17	0.05	1	1	1	1	1	-1
0.40	0.24	0.18	0.18	1	-1	-1	-1	-1	1
0.40	0.24	0.18	0.18	-1	1	1	1	1	-1
0.54	0.10	0.18	0.18	1	-1	-1	-1	-1	1
0.54	0.10	0.18	0.18	-1	1	1	1	1	-1

Table B4: $D-$Optimal OofA Mixture Design Found by TA Algorithm, Example 4.2

x_1	x_2	x_3	x_4	z_{12}	z_{13}	z_{14}	z_{23}	z_{24}	z_{34}
0.40	0.10	0.30	0.20	1	-1	-1	-1	-1	-1
0.40	0.10	0.30	0.20	-1	-1	-1	1	1	-1
0.40	0.10	0.30	0.20	-1	1	1	1	1	1
0.40	0.10	0.30	0.20	1	1	1	-1	-1	-1
0.40	0.10	0.30	0.20	1	1	1	1	1	1
0.80	0.10	0.05	0.05	1	-1	-1	-1	-1	-1
0.80	0.10	0.05	0.05	1	1	-1	1	-1	-1
0.80	0.10	0.05	0.05	-1	-1	-1	-1	-1	1
0.80	0.10	0.05	0.05	1	-1	1	-1	1	1
0.80	0.10	0.05	0.05	-1	1	-1	1	1	-1
0.80	0.10	0.05	0.05	-1	-1	-1	1	1	1
0.80	0.10	0.05	0.05	-1	-1	1	1	1	1
0.80	0.10	0.05	0.05	-1	1	1	1	1	-1
0.80	0.10	0.05	0.05	1	1	1	1	-1	-1
0.80	0.10	0.05	0.05	1	1	1	-1	-1	1
0.80	0.10	0.05	0.05	1	1	1	-1	1	1
0.40	0.50	0.05	0.05	-1	-1	-1	1	-1	-1
0.40	0.50	0.05	0.05	1	1	-1	1	-1	-1
0.40	0.50	0.05	0.05	-1	-1	-1	-1	-1	1
0.40	0.50	0.05	0.05	-1	-1	-1	-1	1	1
0.40	0.50	0.05	0.05	1	-1	1	-1	1	1
0.40	0.50	0.05	0.05	-1	1	-1	1	1	-1
0.40	0.50	0.05	0.05	-1	-1	1	1	1	1
0.40	0.50	0.05	0.05	-1	1	1	1	1	-1
0.40	0.50	0.05	0.05	1	1	1	-1	-1	-1
0.40	0.50	0.05	0.05	1	1	1	-1	-1	1
0.40	0.50	0.05	0.05	1	1	1	1	1	1
0.40	0.25	0.05	0.30	-1	-1	-1	-1	-1	1
0.40	0.25	0.05	0.30	1	-1	1	-1	-1	1
0.40	0.25	0.05	0.30	-1	-1	-1	1	1	-1
0.40	0.25	0.05	0.30	-1	1	-1	1	1	-1
0.40	0.25	0.05	0.30	1	1	1	1	-1	-1
0.40	0.25	0.05	0.30	1	1	1	-1	1	1
0.40	0.25	0.30	0.05	-1	-1	-1	-1	-1	-1
0.40	0.25	0.30	0.05	1	1	-1	-1	-1	-1
0.40	0.25	0.30	0.05	-1	-1	1	-1	1	1
0.40	0.25	0.30	0.05	-1	-1	-1	1	1	1
0.40	0.25	0.30	0.05	1	1	1	1	-1	-1
0.40	0.10	0.20	0.30	1	1	-1	1	-1	-1
0.40	0.10	0.20	0.30	-1	-1	-1	-1	-1	1
0.40	0.10	0.20	0.30	1	-1	1	-1	1	1
0.40	0.10	0.20	0.30	-1	1	1	1	1	-1
0.55	0.10	0.05	0.30	-1	-1	-1	-1	-1	-1
0.55	0.10	0.05	0.30	1	1	-1	1	-1	-1
0.55	0.10	0.05	0.30	-1	-1	-1	-1	1	1
0.55	0.10	0.05	0.30	-1	-1	1	1	1	1
0.55	0.10	0.05	0.30	1	1	1	-1	-1	-1
0.55	0.10	0.05	0.30	1	1	1	1	1	1
0.55	0.10	0.30	0.05	-1	-1	-1	1	-1	-1
0.55	0.10	0.30	0.05	-1	-1	-1	-1	1	1
0.55	0.10	0.30	0.05	1	1	1	-1	-1	1
0.55	0.10	0.30	0.05	1	1	1	1	1	-1
0.58	0.24	0.05	0.13	1	-1	-1	-1	-1	-1
0.58	0.24	0.05	0.13	-1	1	1	1	1	1
0.54	0.24	0.17	0.05	1	-1	1	-1	-1	1
0.54	0.24	0.17	0.05	-1	1	-1	1	1	-1
0.40	0.24	0.18	0.18	1	-1	-1	-1	-1	1
0.40	0.24	0.18	0.18	-1	1	1	1	1	-1
0.54	0.10	0.18	0.18	-1	1	-1	1	-1	-1
0.54	0.10	0.18	0.18	1	-1	1	-1	1	1

Table B5: Summary Statistics of TA Designs for $\{m, l\}$ OofA SLD

m	l	n_i	Mean	SD	Min	Q1	Median	Q3	Max	IQR
4	3	5000	101.25	0.08	100.98	101.20	101.26	101.31	101.38	0.11
		20000	101.25	0.09	100.98	101.21	101.26	101.31	101.38	0.10
		100000	101.26	0.09	100.98	101.21	101.27	101.33	101.38	0.11
4	4	5000	120.14	0.36	119.07	119.72	120.39	120.39	120.39	0.67
		20000	120.26	0.27	119.66	120.39	120.39	120.39	120.39	0.00
		100000	120.26	0.28	119.19	120.39	120.39	120.39	120.39	0.00
6	3	5000	99.73	0.13	99.27	99.65	99.74	99.82	99.99	0.18
		20000	99.84	0.13	99.44	99.78	99.85	99.92	100.17	0.14
		100000	99.86	0.12	99.49	99.81	99.88	99.93	100.16	0.12
6	4	5000	118.03	1.52	111.84	117.07	118.24	119.10	121.21	2.03
		20000	124.15	0.42	123.01	123.87	124.24	124.47	125.00	0.60
		100000	125.12	0.15	124.67	125.05	125.14	125.22	125.40	0.17
6	5	5000	127.20	2.53	121.63	125.64	126.92	128.93	132.93	3.29
		20000	141.67	2.35	135.38	139.99	141.89	143.65	145.94	3.66
		100000	148.70	0.37	146.74	148.52	148.71	148.89	149.57	0.37
8	3	5000	96.29	0.11	96.02	96.21	96.29	96.35	96.51	0.14
		20000	96.64	0.11	96.44	96.57	96.62	96.71	96.94	0.14
		100000	96.76	0.15	96.49	96.66	96.74	96.83	97.52	0.17
8	4	5000	105.03	1.56	101.65	103.84	105.35	106.12	107.92	2.28
		20000	114.73	0.99	111.88	114.23	114.85	115.40	116.43	1.16
		100000	120.48	0.28	119.71	120.32	120.52	120.64	121.06	0.33
8	5	5000	110.93	1.82	107.00	109.67	110.99	111.92	115.76	2.25
		20000	125.17	2.14	120.27	123.60	125.27	126.84	130.79	3.24
		100000	140.01	1.31	136.74	138.98	140.10	140.92	144.12	1.94

Chapter 5
Graphical Approaches to OofA Designs

Existing OofA designs typically assume that all $m!$ orderings of the components are feasible. In many chemical and mixture-type settings, this assumption is not realistic. For example, in the combustion chemical reaction problems examined by Khalila et al. (2017), the reaction is complex and is represented by a graph (or network) with vertices and edges. One example of such a network is shown in Figure 5.1. In Figure 5.1, it is clear that not all chemicals react with each other; e.g., H can react with B, F, or J, but no others. In such networks, it is often desired to find an optimal path that covers all nodes, which is an Order-of-Addition problem, where the experimental region is constrained by a graph.

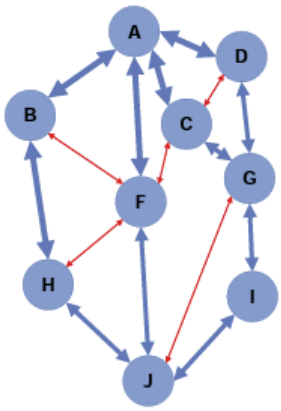


Figure 5.1: A chemical reaction network from Khalila et al. (2017). Vertices represent chemical species, and edges are chemical reactions.

The set of all possible addition orders is also graphically constrained in the field of computer networking. A common problem faced in computer networking is trying to

find a routing path that transmits information to several points with minimum delay, or minimum packet loss (Rusek et al., 2019). Each computer networking problem is characterized by a graph topology. For example, the structure of NSFnet, a network of supercomputers established by the National Research Foundation, is shown in Figure 5.2.

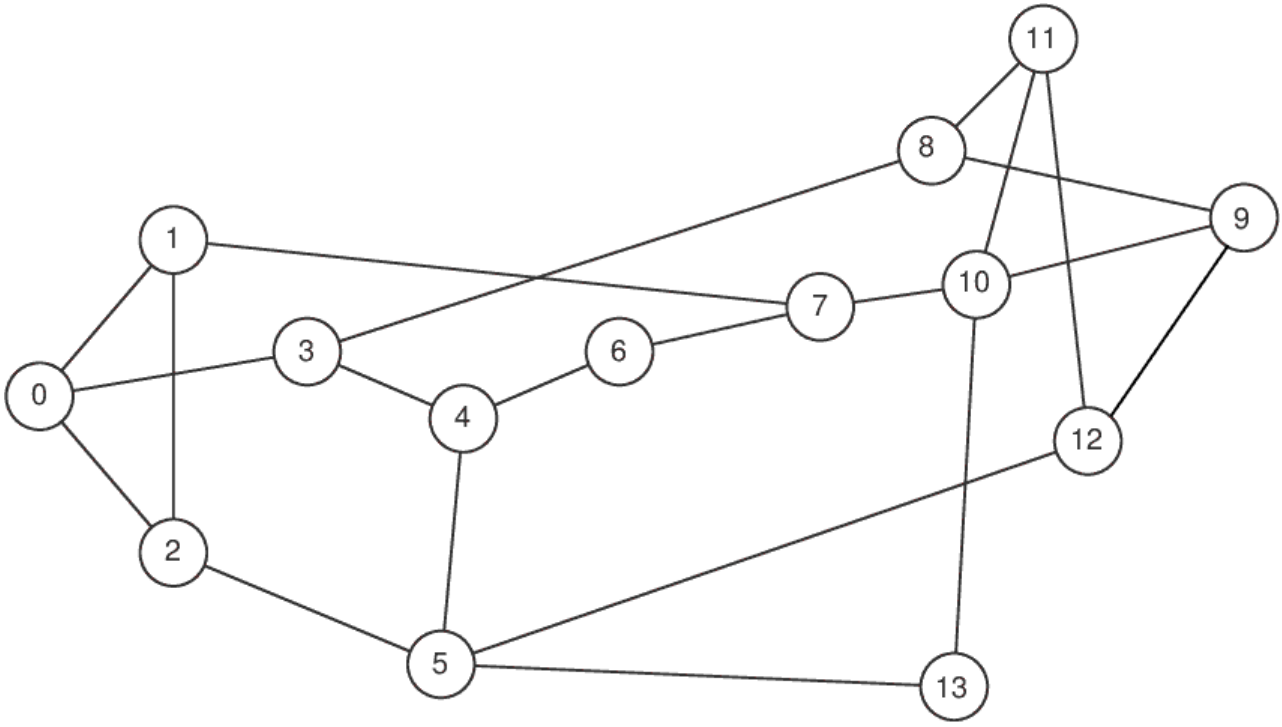


Figure 5.2: The NSFnet topology; figure from Suárez-Varela et al. (2019).

The focus of this chapter is to design optimal OofA experiments when the set of possible addition orders is constrained by an undirected graph. The number of possible orders is bounded above by $m!$, which would correspond to a complete graph, i.e., no restrictions. The existence of constraints inherently reduces the number of feasible runs. Intuitively, this should reduce the sample size required for a good design. However, the designs from Peng et al. (2019) and Chen et al. (2020) may no longer be optimal in the case of graphical constraints, as some of the runs in these designs might not be possible on the graph.

The outline of this chapter is as follows. In Section 5.1, we describe how to represent an OofA experiment as a graph, and we establish theoretical support for estimable and optimal OofA designs under graphical constraints. In Section 5.2, we demonstrate the efficiency of the proposed designs with regard to a class of graphs that support the PWO model. We also explore the distribution of design efficiencies under various graphical models. Section 5.3 provides applications of the methods to real-world examples in

the automotive and pharmaceutical industries. Section 5.3 discusses an extension of the methods in this chapter to a more general OofA Mixture experiment. Section 5.5 concludes the chapter.

5.1 Methods

Let $G = (V, E)$ be an undirected graph with vertex set $V = \{1, 2, \ldots, m\}$ and edge set E, where $E_{ij} = 1$ if it is possible for i and j to be adjacent in at least one feasible permutation; otherwise, $E_{ij} = 0$. Then, the graph G intuitively represents the set of constraints on an OofA experiment; a run of the experiment will be a Hamiltonian path on the graph G, which corresponds to a particular ordering of the m vertices. It is possible to enumerate all feasible permutations by performing a Depth-First Search (DFS) on G. While this is feasible for small m, such a design would require a number of runs of order $O(m!)$. In this section, we explore methods for designing optimal OofA experiments with graphical constraints.

5.1.1 Problem Formulation

Given a graph $G = (V, E)$ that represents the constraints in an OofA experiment, one can use Depth-First Search (DFS) to find the set of all Hamiltonian paths on G. This set of paths is a design D of runs for the OofA experiment. For each path in the design, represent the addition order with the pairwise-ordering (PWO) model (Voelkel, 2019), shown below for convenience:

$$\tau(\mathbf{a}) = \delta_0 + \sum_{jk \in \mathcal{P}} z_{jk}(\mathbf{a})\delta_{jk} \tag{5.1}$$

where $z_{jk}(\mathbf{a}) = 1$ if component j precedes k in path $\mathbf{a}$, and -1 otherwise. In this model, the expected response under the addition order $\mathbf{a}$ is a linear function of the pairwise order of the components. In the context of graphical constraints, the parameters δ_{jk} represent the effect of j preceding k in a Hamiltonian path on G on the expected response $\tau(\mathbf{a})$. In earlier sections, this had a similar interpretation.

There are two questions that naturally arise from this:

1. What conditions on the graph G are sufficient to estimate the parameters in Model (5.1)? Essentially, for any pairwise effect δ_{jk}, what Hamiltonian paths are sufficient to estimate it?

2. Given a set of graphical constraints G, how can an optimal design be found? Specifically, let $\mathcal{D}$ be the set of all design matrices whose rows are Hamiltonian paths on G. Let ϕ be a concave, permutation-invariant optimality criterion under an OofA model f. We want to find

$$D^* = \arg\max_{D \in \mathcal{D}} \phi(M(D))$$

Above, $M(D)$ is the model matrix that corresponds to D under the Model (5.1). Of interest are $D-$ and $A-$ optimality, given by $\phi(M) = |M^T M|$ and $\phi(M) = -tr((M^T M)^{-1})$, respectively. For more details on these criteria, please refer to Section 2.3.

5.1.2 Theoretical Support

Peng et al. (2019) proved that the full OofA design with all $m!$ permutations of $(1, 2, \ldots, m)$ was $\phi-$optimal for all ϕ that were concave and signed permutation invariant. This is based on the assumption that all $m!$ permutations are feasible (i.e. a complete graph). If this is not the case, then any resulting design has a ϕ-optimality that is bounded above by the full design on a complete graph.

Corollary 1. *Let $M_0 = (1/m!) \sum_{\mathbf{a} \in \mathcal{A}} x(\mathbf{a}) x(\mathbf{a})^T$ be the moment matrix that corresponds to an OofA design on a complete graph G. Let G^* be a connected graph formed by removing edges from G, and let M^* be the moment matrix that corresponds to the design of all Hamiltonian paths of G^* used exactly once. Then, for any ϕ that is concave and signed permutation invariant, $\phi(M^*) \leq \phi(M_0)$.*

Corollary 1 establishes the intuitive result that constraints on the set of possible orders may result in a sub-optimal design. It also reinforces the idea that any OofA design based on a graph with m vertices can be compared to the full design on a complete graph as a benchmark. However, not every graph G provides enough Hamiltonian paths to estimate all $\binom{m}{2} + 1$ parameters in the PWO model. As a simple example, consider Figure 5.3, which depicts a cycle graph on four vertices, denoted C_4.

Notice that for any Hamiltonian path $\mathbf{a}$ in Figure 5.3, it is true that $z_{14}(\mathbf{a}) = z_{12}(\mathbf{a}) + z_{23}(\mathbf{a}) + z_{34}(\mathbf{a})$. This implies that it is not possible to estimate δ_{14}, as the resulting model matrix is not full rank. This happens because C_4 is a cycle graph, which forces one pair of effects from $z_{12}(\mathbf{a}), z_{23}(\mathbf{a}), z_{34}(\mathbf{a})$ to have opposite signs for any feasible permutation $\mathbf{a}$ in G. There are other graphs that do not admit PWO estimability. A

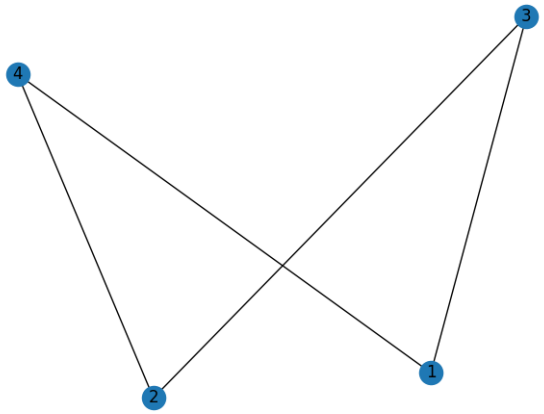


Figure 5.3: The cycle graph C_4.

simple example is given in Lemma 4. Lemma 4 is concerned with bridge edges. These are edges that, if removed, will disconnect the graph into two separate connected components.

Lemma 4. *If a graph G contains a bridge, then G is not PWO estimable.*

What requirements are there for a graph G for full estimability of the $\binom{m}{2}$ PWO effects in Model (5.1)? To answer this, consider how one might estimate the effect δ_{jk} corresponding to $z_{jk}(\mathbf{a})$. A simple way to do this is to use two runs to estimate this effect; one run with j before k, and one where k is before j. In these two runs, all other pairwise orders must be held constant so that the only difference between the two runs is the pairwise order of k and j. This is not the only way to estimate a PWO effect. Lemma 5 generalizes this idea to linear combinations of rows in the model expansion of the design.

Lemma 5. *A PWO effect δ_{jk} from PWO model (5.1) is estimable on G if there exist a set of Hamiltonian paths $\mathcal{H} = \boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell$ in G such that*

$$[c_1 z(\boldsymbol{a}_1) + \cdots + c_\ell z(\boldsymbol{a}_\ell)]^T \delta = \delta_{jk}$$

where $c_1, \ldots, c_\ell$ are real constants, $2 \leq \ell \leq m$, and $z(\boldsymbol{a})$ is a column vector of PWO variables corresponding to the permutation $\boldsymbol{a}$. In particular, if G contains Hamiltonian paths $\boldsymbol{a}_1, \boldsymbol{a}_2$ that satisfy either of the following conditions, the effect δ_{jk} is estimable:

(1) $z_{jk}(\boldsymbol{a}_1) = z_{jk}(\boldsymbol{a}_2)$, but $z_{i\ell}(\boldsymbol{a}_1) = -z_{i\ell}(\boldsymbol{a}_2)$ for all $i, \ell \neq j, k$.

(2) $z_{jk}(\boldsymbol{a}_1) = -z_{jk}(\boldsymbol{a}_2)$, but $z_{i\ell}(\boldsymbol{a}_1) = z_{i\ell}(\boldsymbol{a}_2)$ for all $i, \ell \neq j, k$.

Lemma 5 is a classical but insightful result into how we can estimate effects. As an example, Theorem 1 below shows that a PWO estimable graph can be generated by adding a vertex to a complete graph.

Theorem 1. *Suppose $G = (V, E)$ is a complete graph on m vertices. Let G' be a graph obtained by adding a new vertex $(m+1)$ to G so that it is adjacent to two distinct vertices in V. Then, all $\binom{m+1}{2}$ PWO effects are estimable on G'.*

While Theorem 1 is intuitive, it is restrictive in the sense that $(m-1)$ of the vertices in G must form a complete graph. However, this assumption is not strictly necessary. A complete subgraph was useful in the proof of Theorem 1 because it made it easy to form arbitrary Hamiltonian paths, and it ensured that there was a short path between any two pairs of points. Recall the definition of distance between two vertices in a graph.

Definition 4. *The distance between two vertices j, k, denoted as $d(j, k)$, is the length of the shortest path between j and k in G.*

With this notion of distance in mind, Theorem 2 provides more general conditions that are sufficient for the PWO estimability of G.

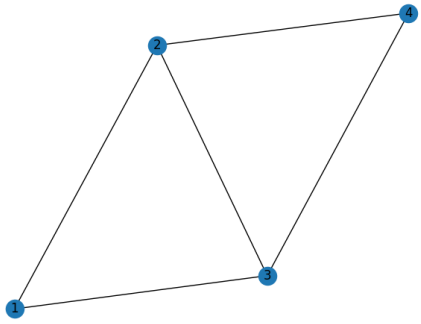
Theorem 2. *Let $G = (V, E)$ be a graph on m vertices. Suppose that:*

1. *For a given pair of adjacent vertices $jk \in V$, there exists $\ell \in V$ that is adjacent to both j and k.*
2. *All adjacent pairs of vertices belong to at least one Hamiltonian cycle in G.*
3. *The maximum distance between any pair of vertices in G is 2.*

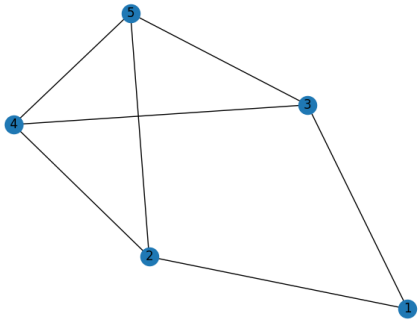
Then, all $\binom{m}{2}$ PWO effects are estimable on G.

Theorem 2 provides three conditions that are sufficient for PWO estimability. Intuitively, it requires that all vertices either belong to a closed triangle or have a short path (of length two) to any other vertex. However, these conditions are not strictly necessary for PWO estimabiltiy. For small m, it is possible to find PWO estimable graphs using computer search. Four such graphs are shown in Figure 5.4.

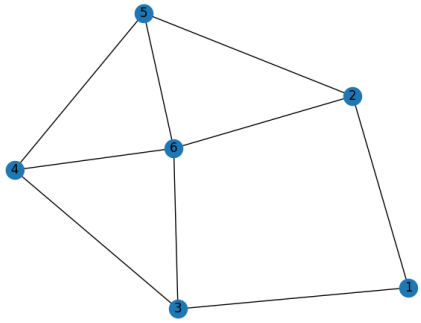
The graphs in Figure 5.4 are "minimal" in the sense that removing any edge from them would render one of the $\binom{m}{2}$ PWO effects δ_{jk} inestimable. For example, in Figure 5.4a, if the edge connecting 2 and 3 is removed, the resulting graph is not estimable, as it is isomorphic to C_4. While it is possible to enumerate these graphs for small values of m, a brute-force approach for large m is computationally inefficient. Instead, we rely on leveraging the conditions of Theorem 2 to produce a known PWO estimable graph.



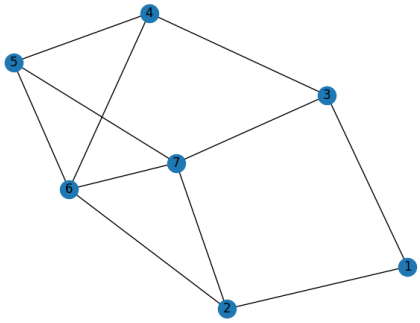
(a) $m = 4$



(b) $m = 5$



(c) $m = 6$



(d) $m = 7$

Figure 5.4: Minimal PWO Estimable Graphs, $m = 4, 5, 6, 7$

5.1.3 Proposed Design

Theorem 2 provides three conditions that are sufficient for the Depth-First Search design on a graph G to support the PWO model. Algorithm 8 shows a simple way to create a graph G that satisfies these conditions. Once a graph that satisfies these conditions is found, it remains to choose a highly efficient subset of all Hamiltonian paths in the graph.

Algorithm 8: Generate PWO Estimable Graph

Initialize $G = K_3$, a triangular graph on 3 vertices.

for $i = 4, 5, \ldots, m$ **do**

Find a pair of vertices $j, k \in V$ with minimal degree.

Add vertex i to the graph. Add edges $(i, j), (i, k), (j, k)$ to the graph.

for $u \in V \setminus \{i, j, k\}$ **do**

If $d(i, u) > 2$, then add an edge from i to u.

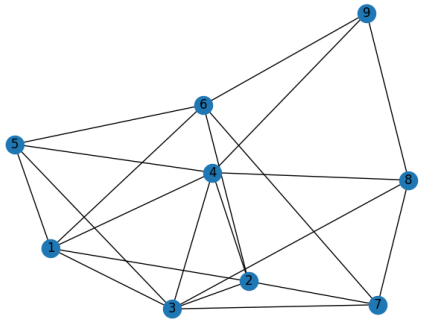
end

end

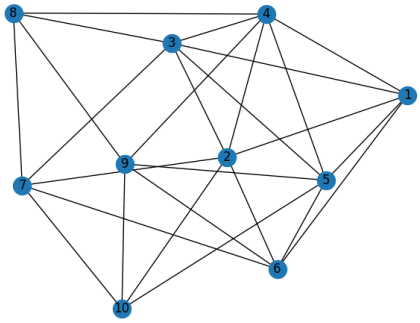
return G

Algorithm 8 begins by creating a triangle graph on 3 vertices. At each iteration in the loop, it identifies two vertices j, k with minimal degree. A new vertex is added to the graph, and edges are added so that it forms a triangle with vertices j and k. This ensures that conditions (1), (2) from Theorem 2 are met. Finally, to satisfy condition (3), the distance between the new vertex and the remaining vertices is checked. If this distance is greater than 2, then an edge is inserted so that the shortest path is of length 1. Figure 5.5, shows the PWO estimable graphs generated by Algorithm 8 are shown for $m = 9, 10, 11, 12$.

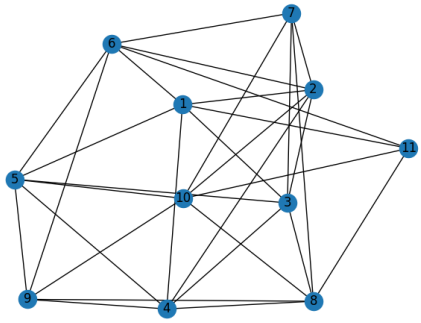
For a general PWO estimable graph G, it is desirable to choose an optimal fraction of the set of all Hamiltonian paths on G. To do this in an efficient way, DFS is used on subgraphs of G that are composed of a half-fraction $s = \lfloor m/2 \rfloor$ of the m vertices of the graph. Intuitively, a single DFS on a subgraph of size s only needs to enumerate at most $s!$ runs, which is computationally more efficient than listing $m!$ runs. Then, a blocking scheme similar to that found in the fractional OofA designs from Peng et al. (2019) were generalized to a graph theoretical framework. This is shown in Algorithm 9 for an even number of components m.



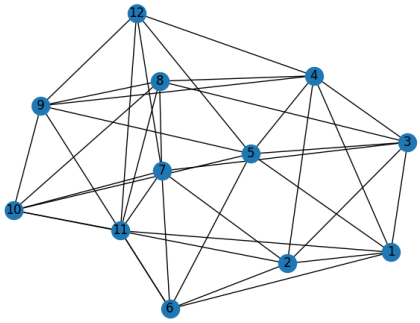
(a) $m = 9$



(b) $m = 10$



(c) $m = 11$



(d) $m = 12$

Figure 5.5: PWO Estimable Graphs from Algorithm 8, $m = 9, 10, 11, 12$

Algorithm 9: Find Fractional DFS Design

Input: A graph G with m vertices. Let $s = \lfloor m/2 \rfloor$. Initialize a design matrix D.

Find all subgraphs of G with s vertices, and arrange them in lexicogrpahical order. Denote these as $G_u, u = 1, 2, \ldots, L$, and their complements as $\bar{G}_u$.

for $u = 1, 2, \ldots, L$ **do**

$B_u = \text{DFS}(G_u)$, $\bar{B}_u = \text{DFS}(\bar{G}_u)$

Let $\sim \bar{B}_u$ be the column reversal of $\bar{B}_u$.

for r = *1,2,...,* $nrow(B_u)$ **do**

If $B_u[r, s]$ is adjacent to $\bar{B}_u[r, 1]$, then concatenate these two rows and add them to the design D.

If $\sim \bar{B}_u[r, s]$ is adjacent to $B_u[r, 1]$, then concatenate these two rows and add them to the design D.

end

end

return D

Algorithm 9 finds all subgraphs of size $s = \lfloor m/2 \rfloor$ vertices of a given graph G. These are arranged in lexicographical order. On each subgraph (and its complement), DFS is used to list all permutations of the s components in lexicographical order. This makes the design easier to enumerate, as there are at most $s!$ permutations to be listed by a DFS on s vertices. These groups of permutations form blocks $B_u, \bar{B}_u$ for $u = 1, 2, \ldots, L$, where L is the total number of blocks. Unlike the algorithm in Peng et al. (2019), the blocks can not simply be joined. Each pair of rows must be checked to ensure that, if concatenated forms a path in G. If m happens to be odd, then some extra care is taken to find an efficient fraction of runs. In Peng et al. (2019), the optimal design for odd m is found by stacking m copies of D, and then inserting a column of $m's$ before the ℓ^{th} row of the ℓ^{th} copy. In this case, such an insertion may only be done if vertex m is adjacent to both of the vertices in positions $\ell - 1, \ell + 1$ of each row of the ℓ^{th} copy of D.

5.2 Design Efficiency Under Simulated Random Graphs

In this section, we compare the relative $D-$ and $A-$ efficiencies of the proposed design for many graphs. For an optimality criterion ϕ, the relative $\phi-$efficiency of a moment matrix M_1 with respect to M_2 is $\frac{\phi(M_1)}{\phi(M_2)}$. To find the relative $D-$efficiency of a moment matrix M to the corresponding full moment matrix M_{full}, this ratio is $\frac{\phi(M)}{\phi(M_{full})}$ (as larger $D-$efficiency is better). $A-$efficiency measures the average variance, so smaller values of $A-$efficiency are better. Therefore, for $A-$efficiency, this ratio is inverted. In this context, the full moment matrix M_{full} corresponds to the full DFS design on all possible paths in a graph G.

It is of interest to see how these designs hold under general graphical constraints. To examine behavior of these designs under "typical" graphs, random graphs are employed. In random graph theory, a probability distribution is placed over a set of graphs (Frieze and Karoński, 2016). Since the focus is on the OofA problem, the set of graphs is restricted to those that already contain at least one Hamiltonian path $(1, 2, \ldots, m)$; otherwise, no resulting design is possible. A modified Binomial random graph model is used, where each of the additional edges has constant probability p of being included (Gilbert, 1959). By examining relative $D-$ and $A-$ efficiencies for varying levels of p, it is possible to infer how these designs perform for different levels of sparsity in a graph.

Figure 5.6 shows boxplots the relative $D-$ and $A-$ efficiencies of the fractional designs over $N = 100$ modified $\mathcal{G}(m, p)$ random graphs for $m = 6, 7, 8, 9$. These efficiencies are relative to the theoretical complete design with $m!$ runs. For each m, 100 simulations were

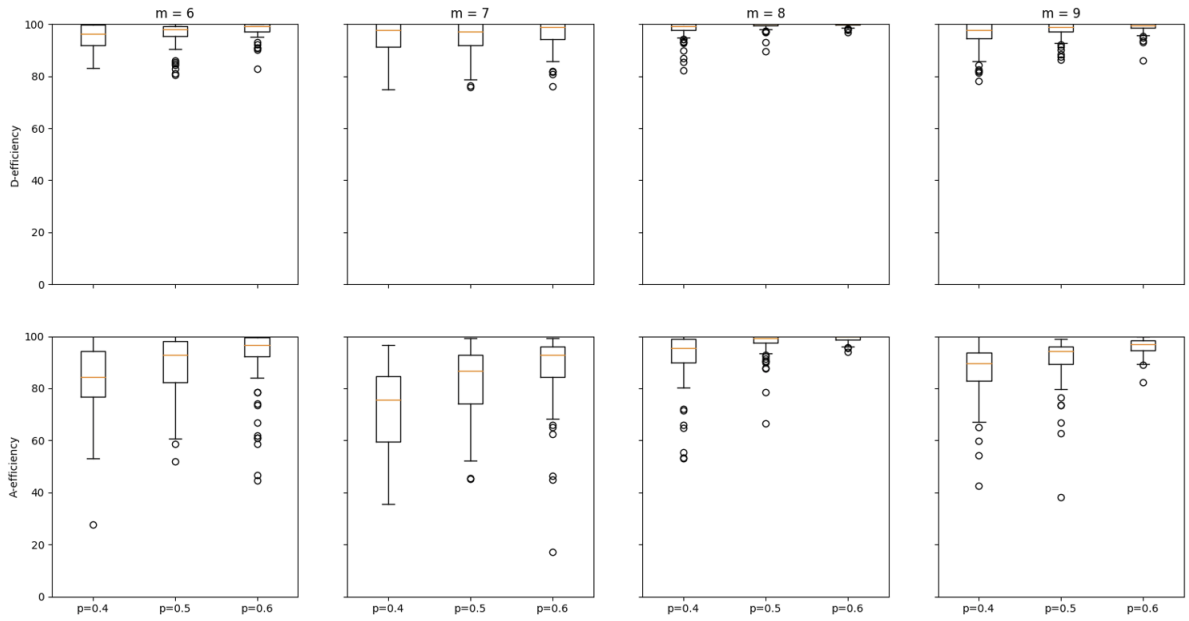


Figure 5.6: Relative $D-$ and $A-$ efficiencies for $N = 100$ Binomial Random Graphs

run for each $p = 0.4, 0.5, 0.6$. Overall, as the proportion of edges in the graphs increase, the minimum relative $D-$efficiency increases, and the spread (i.e. IQR) decreases. This indicates that more edges in general lead to more optimal designs, which is intuitive. As m increases, the relative efficiencies also increase and have lower IQR, which is very desirable. over half of the random designs had relative $D-$ efficiency above 80% for $m = 7, 8, 9$. The $A-$efficiency exhibits more variability for $m = 6, 7$, but for smaller values of m, it is easier to find an optimal design using a greedy search algorithm over the set of Hamiltonian paths. Overall, Figure 5.6 suggests that the design approach is robust for large m, e.g. $m \geq 8$.

One criticism of the $\mathcal{G}(m, p)$ model is that it does not reflect the natural clustering that occurs in real-world networks. In many real applications, such as social networks, the degree distribution of the vertices tends to follow a skewed distribution or power law. To address this, the Barabasi-Albert (BA) model was used to generate random graphs (Barabási and Albert, 1999). In this model, new vertices are added an initial completely connected network of $m_0 < m$ vertices. The probability that a new vertex forms an edge with an existing vertex v is directly proportional to the degree of v. In this application, three edges were added for each new vertex. For more details on the Python implementation of this model, see Hagberg et al. (2008). In Figure 5.7, the distributions for the relative $D-$efficiencies of these designs are shown for $m = 6, 7, 8, 9$

and $m_0 = 3, 4, 5$ vertices in the initial network. Overall, we see that the median relative $D-$ and $A-$efficiencies to the full DFS design are above 80% for $m = 8, 9$ for each choice of the initial amount of vertices m_0. For $m = 6, 7$, there appears to be more variability in the relative $A-$efficiency in terms of IQR, while the relative $D-$efficiencies are clustered around 90%.

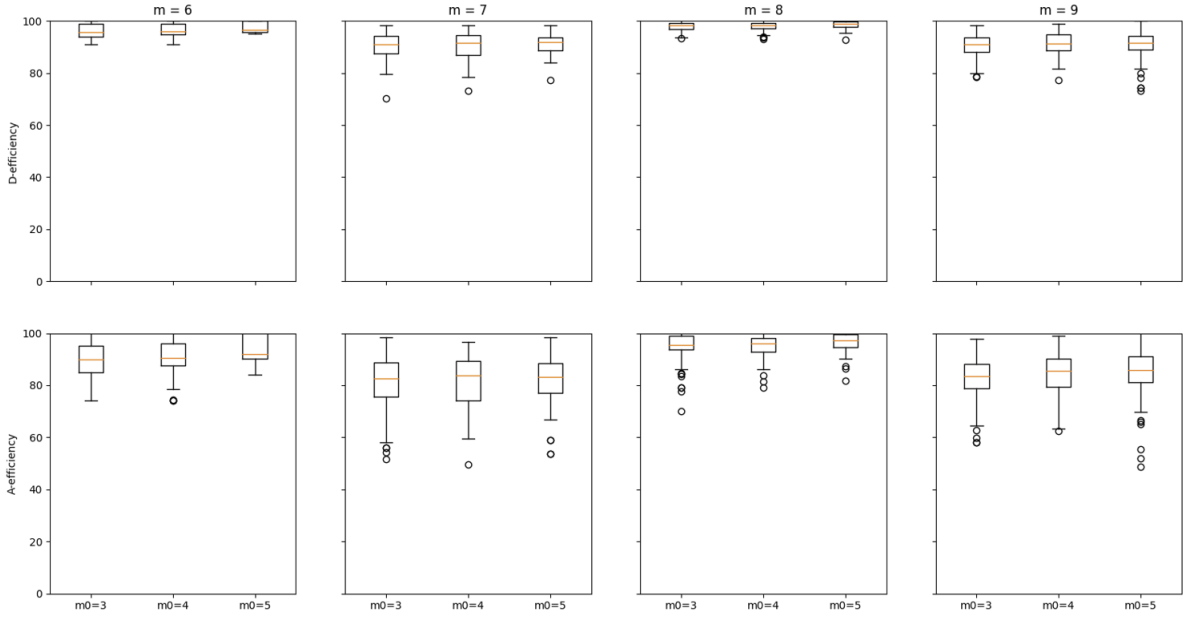


Figure 5.7: Relative $D-$ and $A-$ efficiencies for $N = 100$ BA Random Graphs

5.3 Examples

In this section, graphical constraints are applied to two real-world examples. In each example, the response depends on the order of addition of several components. The methods described in Section 5.1 are applied to fit PWO models to the datasets under graphical constraints, and optimal orders are found.

5.3.1 Automotive Coating

As an example, we consider data from Voelkel and Gallagher (2019) on the viscosity of paint in an automotive coating system. The order of $m = 4$ components (two binder resins, a flow and leveling additive, and a rheology modifier) were varied, and their impact on the viscosity of the coating was measured. The viscosity was recorded as a function of shear rate (measured in s^{-1}). In particular, viscosity at a low level (LSV) and a high

level (HSV) were recorded. According to Voelkel and Gallagher (2019), it is desired to have the ratio LSV/HSV be large, as this implies that the visocity will decrease with increased force of spraying or painting. The provided dataset records this ratio on the log scale.

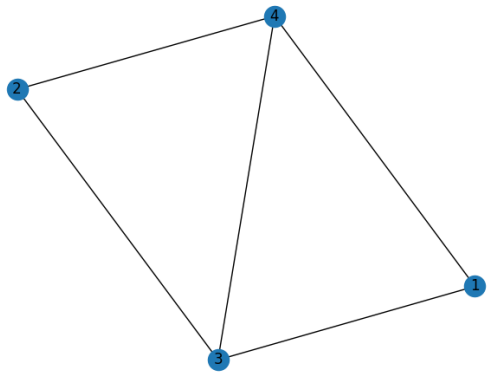


Figure 5.8: Graph Generated by Algorithm 8 for $m = 4$

We apply Algorithm 8 to find a PWO estimable graph with $m = 4$ components. This is equivalent to preventing components 1 and 2 from being run sequentially. In this case, the experiment is constrained by a graph that is isomorphic to the graph in Figure 5.8. This is also a minimal estimable PWO graph, so further edges cannot be removed. Model (5.1) was fit to the data and used search for an addition order that maximizes the log(LSV/HSV) ratio. This model was fit to all runs in the Fractional DFS design.

Table 5.1: Parameter Estimates for PWO Model

Parameter	Estimate	SE	t	p-value
δ_0	0.2618	0.012	21.688	<0.0001
δ_{12}	-0.0386	0.036	-1.065	0.335
δ_{13}	-0.0002	0.021	-0.010	0.992
δ_{14}	-0.2028	0.021	-9.699	<0.0001
δ_{23}	-0.0149	0.021	-0.714	0.507
δ_{24}	0.0113	0.021	0.542	0.611
δ_{34}	-0.0126	0.021	-0.605	0.572

As shown in Table 5.1, the only significant parameter is δ_{14} at the $\alpha = 0.05$ level. Since $\hat{\delta}_{14} = -0.2028$ has a negative sign, this indicates that component 1 should be placed after component 4 in the optimal ordering to get a larger response. Therefore, the potential optimal orderings are the set of Hamiltonian paths that do not contain the edge

$(1, 2)$ and have component 1 after component 4. The fitted model was used to predict the response for every path that met these criterion, and it was found that the optimal path was $\mathbf{a}^* = (3, 2, 4, 1)$ with a predicted maximum of $\hat{y}^* = 0.517$. When consulting the data, the true optimal order is indeed $(3, 2, 4, 1)$ with a response of $y^* = 0.537$.

In the example above, the edge $(1, 2)$ was removed. There are actually six possible edges that could have been removed to generate a minimal PWO estimable graph. The PWO model was fit to the data for each of these cases, and the optimal order was identified by examining the fitted responses. In Table 5.2, we show the true top five orders, according to the response, with the order predicted by the PWO model in bold. In all but one case, the PWO model identified the optimal order. When the edge $(2, 4)$ was removed, the PWO model identified the second best order. Overall, the model performs well in all six cases.

Table 5.2: Top Five Resin Orders for each Removed Edge

Edge Removed	Order				Response	Edge Removed	Order				Response
(1,2)	**3**	**2**	**4**	**1**	0.537	(2,3)	**2**	**4**	**3**	**1**	0.534
	2	4	3	1	0.534		3	4	2	1	0.502
	2	3	4	1	0.510		2	4	1	3	0.482
	4	2	3	1	0.499		4	3	1	2	0.476
	2	4	1	3	0.482		4	2	1	3	0.436
(1,3)	**3**	**2**	**4**	**1**	0.537	(2,4)	2	3	4	1	0.510
	2	3	4	1	0.510		**4**	**3**	**2**	**1**	0.491
	3	4	2	1	0.502		4	3	1	2	0.476
	4	3	2	1	0.491		4	1	2	3	0.461
	4	1	2	3	0.461		3	4	1	2	0.414
(1,4)	**2**	**4**	**3**	**1**	0.534	(3,4)	**3**	**2**	**4**	**1**	0.537
	3	4	2	1	0.502		4	2	3	1	0.499
	4	2	3	1	0.499		2	4	1	3	0.482
	4	3	2	1	0.491		4	1	2	3	0.461
	4	3	1	2	0.476		4	2	1	3	0.436

Predicted optimal orders are indicated in **bold**.

5.3.2 Combinatorial Drug Therapy

For a second example, data on combinatorial drug therapy from Yang et al. (2021) is used. There are $m = 4$ chemotherapeutics used to treat lymphoma. The objective of the study is to identify an addition order of these treatments that maximizes the cell inhibition (y). The cell inhibition was measured half a day after the final drug in the

sequence was administered. To illustrate the proposed methods, we consider imposing all possible PWO estimable graphs on $m = 4$ vertices. As in the previous example, there are six possible graphs to consider. A fractional DFS design with $n = 12$ runs was found for each graph. The PWO model was fit to each of these designs. The same procedure from the previous example was repeated to predict the optimal orders. Each of these graphs corresponds to a removal of one edge from the complete graph K_4. The optimal orderings found by the PWO model are reported in Table 5.3.

Table 5.3: Top Five Drug Orders for Each Removed Edge

Edge Removed	Order				Response	Edge Removed	Order				Response
(1,2)	**1**	**3**	**4**	**2**	56.5	(2,3)	1	3	4	2	56.5
	1	3	2	4	55.4		3	1	2	4	53.5
	3	2	4	1	51.4		3	4	2	1	53.4
	1	4	3	2	51.2		**3**	**4**	**1**	**2**	52.9
	3	1	4	2	51.2		3	1	4	2	51.2
(1,3)	3	4	1	2	53.4	(2,4)	**3**	**4**	**1**	**2**	52.9
	3	**4**	**1**	**2**	52.9		1	4	3	2	51.2
	3	2	4	1	51.4		3	2	1	4	50.8
	1	4	3	2	51.2		4	3	1	2	46.8
	3	2	1	4	50.8		2	3	4	1	46.4
(1,4)	**1**	**3**	**4**	**2**	56.5	(3,4)	1	3	2	4	55.4
	1	3	2	4	55.4		3	1	2	4	53.5
	3	1	2	4	53.5		3	2	4	1	51.4
	3	4	2	1	53.4		**3**	**1**	**4**	**2**	51.2
	4	3	1	2	46.8		3	2	1	4	50.8

Predicted optimal orders are indicated in **bold**.

Table 5.3 shows the top five true addition orders for the drugs in each of the six scenarios where an edge was removed from the graph. The optimal ordering predicted by the PWO model is indicated in bold. In all cases, the ordering predicted by the PWO model is one of the top five orders. In two thirds of the cases, it is one of the top two orders. The removal of edges $(2, 3)$ and $(3, 4)$ resulted in the least optimal predicted orders. To see why, the PWO model was fit to the data without any graphical constraints. In this case, the significant PWO parameters were δ_{23} ($T = -3.742$, p-value 0.002) and δ_{34} ($T = 4.115$, p-value 0.001). This suggests that removing the edges corresponding to these effects makes it more difficult to identify the optimal order.

5.4 Extension to OofA Mixture Experiments

So far, we have only examined changing the order of components in a mixture-type experiment with the ratios of the components held constant. As previous chapters have stated, this approach does not yield any knowledge about the mixture proportions or their interactions with the addition order. It is of interest to design OofA Mixture experiments in the presence of graphical constraints on the order of addition. Fortunately, the graphical approach discussed in this chapter has a natural extension to the case of the OofA Mixture experiment.

Suppose we wish to create a design based on t mixture points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t$. First, construct a graph with vertex set $V = \{1, 2, \ldots, m, \mathbf{x}_1, \ldots, \mathbf{x}_t\}$. The subgraph induced by the vertices $\{1, 2, \ldots, m\}$ will represent the graphical constraints on the order of addition. For each mixture point $\mathbf{x}$, draw a directed edge from $\mathbf{x}$ to each $j \in \{1, 2, \ldots, m\}$ such that the j^{th} element of $\mathbf{x}$ is nonzero. An example of this graph is given for the $\{3, 2\}$ OofA SLD in Figure 5.9.

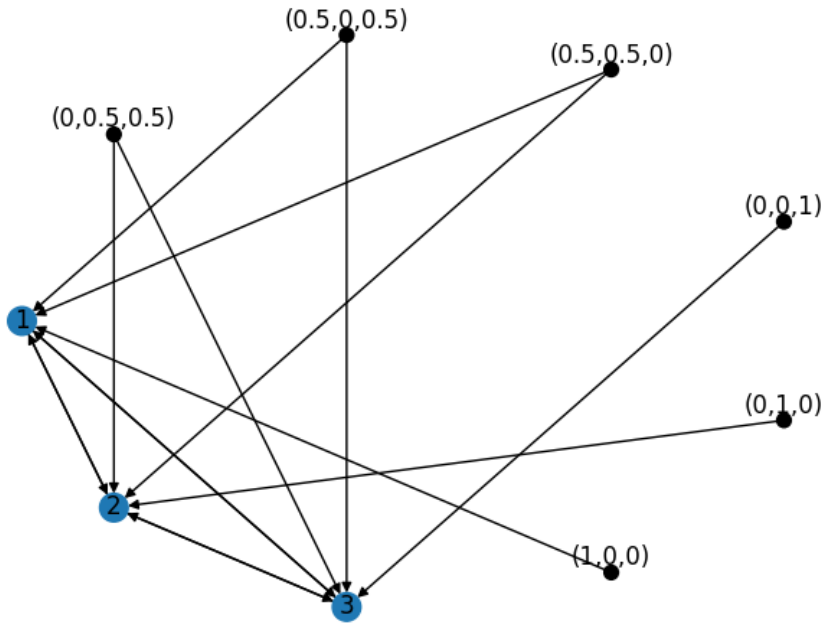


Figure 5.9: Graphical Representation of the $\{3, 2\}$ OofA SLD.

With a graphical representation of the OofA Mixture experiment created, Algorithm 10 shows how to traverse the graph to create the corresponding OofA Mixture design.

Algorithm 10: Find OofA Mixture Design with DFS

Input: A graph G with vertex set $V = \{1, 2, \ldots, m, \mathbf{x}_1, \ldots, \mathbf{x}_t\}$ that represents an OofA Mixture experiment.

Initialize a design matrix D.

for $i = 1, 2, \ldots, t$ **do**

Let G_i be the subgraph of G induced by the vertices $(\mathbf{x}_i, \mathcal{N}(\mathbf{x}_i))$.

Let k be the number of nonzero elements of $\mathbf{x}_i$.

Let P_i be the set of all paths of the form $(\mathbf{x}_i, \pi_1, \ldots, \pi_k)$ found from a DFS (or Fractional DFS) of G_i starting at root node $\mathbf{x}_i$.

Let D_i be the matrix that codes all permutations in P_i via modified PWO variables. Store D_i.

end

return $D = [D_1^T, \ldots, D_t^T]^T$

As an input, Algorithm 10 takes a graphical representation of an OofA Mixture design. The algorithm creates one induced subgraph G_i for each mixture $\mathbf{x}_i, i = 1, \ldots, t$. This subgraph consists of the vertices $\mathbf{x}_i$ and the components j such that the j^{th} element of $\mathbf{x}_i$ is nonzero. Since the edge from $\mathbf{x}_i$ to each j is directed, a DFS on G_i rooted at $\mathbf{x}_i$ will always produce paths of the form $(\mathbf{x}_i, \pi_1, \ldots, \pi_k)$ where k is the number of nonzero elements of $\mathbf{x}_i$, and $(\pi_1, \ldots, \pi_k)$ is a permutation of the indices $\{j \mid x_{ij} \neq 0\}$. Algorithm 10 then codes the permutations using the modified PWO variables, juxtaposes them with $\mathbf{x}_i$, and stores them in a matrix D_i. The final design is found by stacking the rows of $D_i, i = 1, 2, \ldots, t$.

Algorithm 10 is a generalization of Algorithm 2. If the subgraph induced by $\{1, 2, \ldots, m\}$ on G is the complete graph K_m, then the resulting OofA Mixture designs will be identical. The advantage of Algorithm 10 is that it is applicable if there are restrictions on the addition order of the components. In either case, we can reduce the total number of runs by replacing the DFS procedure with the fractional DFS procedure provided in Algorithm 9.

5.5 Conclusion

To the best of our knowledge, this is the first attempt to provide guidance on how to design Order-of-Addition (OofA) experiments on a pre-existing graph. Although OofA designs have received much recent attention in recent years, existing attempts at finding optimal or efficient designs have assumed that all $m!$ runs are feasible in the

experiment (Chen et al., 2020; Lin and Peng, 2019; Zhao et al., 2022). The proposed theory establishes a more general framework for estimability of pairwise-ordering (PWO) components in OofA experiments by restricting the focus of the design space to the set of all Hamiltonian paths on a graph. This provides key insight into which types of constraints in an OofA experiment allow for the popular PWO model to be used.

Several contributions to the OofA experiment have been made here. First and foremost, we provide a natural framework for the practical case where constraints are placed on the set of all $m!$ runs. Theory is provided for assessing the estimability of the parameters in the popular PWO model for a fixed graph. Sufficient and reasonable conditions for model estimability are provided. Moreover, given a PWO estimable graph, designs are proposed that are highly efficient when compared to the design that results from listing all permutations on the graph. Simulations show that for random grahical constraints generated by the Binomial and Barabasi-Albert models, these designs have high efficiency relative to the full design with all $m!$ runs when m is large (e.g. $m \geq 8$).

The designs proposed in this chapter are flexible, and apply to several relevant applications. Additional methodology is also proposed that makes this approach compatible with the OofA Mixture problem. Many modern Order-of-Addition problems involve graphical constraints on the set of possible orders. In food science and chemistry, mixtures often require that certain ingredients not be added in direct sequence. In computer science, graphs appear in social networks of users, computer networks, and many other areas. Although this problem appears in many fields, there has been a lack of research into how to design OofA experiments under these constraints. It is our hope that this research will open up new applications of OofA designs.

5.A Appendix: Proofs

Proof of Corollary 1. Let $\mathcal{A}^*$ be the set of all Hamiltonian paths in G^*. Define

$$w^*(\mathbf{a}) = \begin{cases} 1/|\mathcal{A}^*| & \text{if } \mathbf{a} \in \mathcal{A}^* \\ 0 & \text{otherwise} \end{cases}$$

By definition of w^*, $M(w^*) = \sum_{\mathbf{a}\in\mathcal{A}} w^*(\mathbf{a})x(\mathbf{a})x(\mathbf{a})^T = (1/|\mathcal{A}^*|)\sum_{\mathbf{a}\in\mathcal{A}^*} x(\mathbf{a})x(\mathbf{a})^T = M^*$. Since $w^*(\mathbf{a}) \geq 0$ for any $\mathbf{a} \in \mathcal{A}$, and $\mathcal{A}^* \subset \mathcal{A}$, then w^* is a design measure on the set $\mathcal{A}$. The result follows from Theorem 1 of Peng et al. (2019). □

Proof of Lemma 4. Let e_{jk} be a bridge in G for vertices j, k, where $j < k$. By

definition, removing e_{jk} from the graph will disconnect the graph, creating two connected components G_1, G_2, with $j \in G_1, k \in G_2$. Furthermore, any Hamiltonian path $\mathbf{a} \in G$ must cross the bridge e_{jk}. Therefore, for any vertices $u \neq j$ in G_1 and $v \neq k$ in G_2, we have that $z_{jk}(\mathbf{a}) = z_{uv}(\mathbf{a})$ for any Hamiltonian path $\mathbf{a}$ in G. Hence, the resulting PWO model matrix cannot be full rank. □

Proof of Theorem 1. Let $G = (V, E)$ be a complete graph. Let G' be the graph formed by adding a vertex $m+1$ and making it adjacent to two distinct vertices in G. Without loss of generality, suppose $m+1$ is adjacent to vertices 1 and 2 (if this is not the case, simply re-label the vertices in the complete graph). Since G is complete and G' forms a closed triplet with $1, 2$, then the following Hamiltonian paths exist in G':

$$\begin{aligned}
\mathbf{a}_{1,m+1,1} &= [m+1, 1, 2, 3, \ldots, m] \\
\mathbf{a}_{1,m+1,2} &= [1, m+1, 2, 3, \ldots, m] \\
\mathbf{a}_{2,m+1,1} &= [2, m+1, 1, 3, \ldots, m] \\
\mathbf{a}_{2,m+1,2} &= [m+1, 2, 1, 3, \ldots, m]
\end{aligned}$$

Notice that $\mathbf{a}_{1,m+1,1}, \mathbf{a}_{1,m+1,2}$ only differ in the pairwise ordering of $1, m+1$, and $\mathbf{a}_{2,m+1,1}, \mathbf{a}_{2,m+1,2}$ only differ in the pairwise ordering of $2, m+1$. By Lemma 5, $\delta_{1,m+1}$ and $\delta_{2,m+1}$ are estimable. For any pair jk such that $3 \leq j < k < m+1$, there exist a pair of Hamiltonian paths in G' of the form:

$$\begin{aligned}
\mathbf{a}_{j,k,1} &= [j, k, 2, 3, \ldots, 1, m+1] \\
\mathbf{a}_{j,k,2} &= [k, j, 2, 3, \ldots, 1, m+1]
\end{aligned}$$

We also note that there are paths that allow estimation of δ_{12}:

$$\begin{aligned}
\mathbf{a}_{1,2,1} &= [m+1, 1, 2, 3, \ldots, 1, m] \\
\mathbf{a}_{1,2,2} &= [m+1, 2, 1, 3, \ldots, 1, m]
\end{aligned}$$

It is also possible to estimate δ_{1k} for any $2 < k < m+1$:

$$\begin{aligned}
\mathbf{a}_{1,k,1} &= [1, k, 3, \ldots, 2, m+1] \\
\mathbf{a}_{1,k,2} &= [k, 1, 3, \ldots, 2, m+1]
\end{aligned}$$

Similarly, exchanging 1 and 2 above shows that δ_{2k} is estimable for $2 < k < m+1$.

Therefore, by Lemma 5, δ_{jk} is estimable for any $1 \leq j < k < m+1$. It remains to show that $\delta_{k,m+1}$ is estimable for any $2 < k \leq m$. Consider the following Hamiltonian paths:

$$\mathbf{a}_{k,m+1,1} = [m+1, 1, k, 2, \ldots, m]$$
$$\mathbf{a}_{k,m+1,2} = [k, 1, m+1, 2, \ldots, m]$$

Notice that

$$\begin{aligned}(z(\mathbf{a}_{k,m+1,1}) - z(\mathbf{a}_{k,m+1,2}))^T\delta &= 2\delta_{1k} - 2\delta_{1,m+1} - 2\delta_{k,m+1}\\ (z(\mathbf{a}_{1,k,1}) - z(\mathbf{a}_{1,k,2}))^T\delta &= 2\delta_{1k}\\ (z(\mathbf{a}_{1,m-1,1}) - z(\mathbf{a}_{1,m-1,2}))^T\delta &= -2\delta_{1,m-1}\end{aligned}$$

Therefore,

$$-0.5\Big(z(\mathbf{a}_{k,m+1,1}) - z(\mathbf{a}_{k,m+1,2}) - z(\mathbf{a}_{1,k,1}) + z(\mathbf{a}_{1,k,2}) - z(\mathbf{a}_{1,m-1,1}) + z(\mathbf{a}_{1,m-1,2})\Big)^T\delta = \delta_{k,m+1}$$

Hence, a linear combination of rows of the design matrix exists that satisfies the condition in Lemma 1 for $\delta_{k,m+1}$. The concludes the proof. □

Proof of Theorem 2. Let $j < k$. There are two cases.

(a) Vertices j and k are adjacent. Then there exists $\ell \in V$ that are adjacent to both j and k. By assumption, jk belongs to a Hamiltonian cycle in G. Then, the Hamiltonian paths

$$\mathbf{a}_1 = [j, k, \ell, \pi_1, \ldots, \pi_m]$$
$$\mathbf{a}_2 = [k, j, \ell, \pi_1 \ldots, \pi_m]$$

also exist in G. Then by (1) in Lemma 5, δ_{jk} is estimable.

(b) Vertices j and k are not adjacent. Since $\max_{u,v\in V} d(u,v) = 2$, then it must be the case that $d(j,k) = 2$. Therefore, there exists some ℓ that satisfies $d(j,\ell) = d(\ell,k) = 1$; i.e., j, k share a common neighbor ℓ. By case (a), $\delta_{j\ell}, \delta_{\ell k}$ are directly estimable by using the paths:

$$\mathbf{a}_{j\ell 1} = [j, \ell, \pi_1, \ldots, k, \ldots, \pi_m]$$
$$\mathbf{a}_{j\ell 2} = [\ell, j, \pi_1 \ldots, k, \ldots, \pi_m]$$

$$\mathbf{a}_{\ell k1} = [\ell, k, \pi_1 \ldots, j, \ldots, \pi_m]$$
$$\mathbf{a}_{\ell k2} = [k, \ell, \pi_1 \ldots, j, \ldots, \pi_m]$$

Let $\mathbf{a}_{jk1} = [\pi_1, \ldots, j, \ell, k, \ldots, \pi_m]$ and $\mathbf{a}_{jk2}$ denote the same path, but with k and j exchanged. Then, it follows that

$$(z(\mathbf{a}_{jk1}) - z(\mathbf{a}_{jk2}))^T \delta = 2\delta_{jk} + 2\delta_{j\ell} + 2\delta_{\ell k}$$
$$(z(\mathbf{a}_{\ell k1}) - z(\mathbf{a}_{\ell k2}))^T \delta = 2\delta_{\ell k}$$
$$(z(\mathbf{a}_{j\ell 1}) - z(\mathbf{a}_{j\ell 2}))^T \delta = 2\delta_{j\ell}$$

Therefore,

$$0.5\Big(z(\mathbf{a}_{jk1}) - z(\mathbf{a}_{jk2}) - z(\mathbf{a}_{\ell k,1}) + z(\mathbf{a}_{\ell k2}) - z(\mathbf{a}_{j\ell 1}) + z(\mathbf{a}_{j\ell 2})\Big)^T \delta = \delta_{jk}$$

In both cases, it is possible to estimate δ_{jk}. □

Chapter 6
Conclusions and Future Work

6.1 Conclusions

There are many mixture experiments where the response depends on the order in which components are added. Prior to this research, little attention was taken towards designing an experiment where both of these factors could be simultaneously analyzed. Mee (2020) emphasized this, stating that including mixture proportions is a "natural extension" of the OofA problem. In this dissertation, a solid framework for designing and analyzing Order-of-Addition Mixture Experiments has been established. An algorithm for enumeration of full OofA Mixture designs has been proposed. It was proven that these designs ensure that order and mixture effects are orthogonal, which is very helpful result for ANOVA. Models that include interaction effects between mixture proportions and modified PWO variables were proposed. It was demonstrated via simulation and examples that when these effects are present, the optimal mixture proportions detected by a pure mixture model may be misleading. Much attention was also given to finding the optimal mixture proportions and order. When m is large, a Simulated Annealing algorithm was proposed to find the optimal inputs, and its efficacy was demonstrated in examples.

One of the largest barriers to the simultaneous analysis of mixture proportions and order of addition is the presumably large sample size required by traditional approaches. Another major contribution of this work was the construction of highly efficient small-run subsets of the full OofA Mixture design. The TA algorithm was adapted to this problem to find affordable OofA Mixture designs under the $D-$ and $I-$optimality criteria. $D-$optimal designs are very desirable for statistical inference, and they are often sought in the OofA literature. On the other hand, $I-$optimal designs are incredibly useful for prediction, and they are desirable in mixture experiments because the aim is to estimate the response surface and find the optimal mixture proportions. We find a closed form for

the $I-$efficiency under the additive OofA Mixture model, and we propose a Monte Carlo approach for estimating this quantity in a general scenario. These methods can also be employed when there are single-component constraints on the mixture proportions, which is a common problem in mixture experiments. Many of these designs have been found to have higher $D-$ or $I-$efficiency relative to the full OofA Mixture design, which provides evidence that it is not necessary to use all $t \times m!$ runs in an experiment with t mixtures. This will enable researchers in the chemical, pharmaceutical, and biological engineering fields to search for optimal order and mixture proportions at a low cost.

Finally, efficient Order-of-Addition designs are proposed for problems where the set of feasible orders is constrained by a network. This is often a problem of interest in mixture experiments or combinatorial drug therapy, where certain chemical reagents or drugs cannot be taken in direct sequence. These methods are also useful in wireless computer networking and job scheduling. Sufficient theoretical conditions on a graph are derived for estimability of all $\binom{m}{2}$ pairwise order effects of the components. A graph is shown to impose an OofA design via a Depth-First Search traversal. A fractional DFS algorithm is used to reduce the number of runs in the constrained OofA experiment. Simulations show that for large m, these fractional DFS designs maintain relatively high $D-$ and $A-$efficiency under many simulated random graphs. Applications to combinatorial drug therapy and automotive paint coating are shown, and the PWO model performs well in terms of identifying optimal orders. Furthermore, a general graph-based algorithm is provided for enumerating the runs in an OofA Mixture experiment when graphical constraints are present. This not only demonstrates that OofA Mixture experiments can be designed under graphical constraints, but it also emphasizes the flexibility of a graph-theoretic approach.

6.2 Future Work

There is still room for development in OofA Mixture experiments. Additional approaches to modeling can be explored, particularly in the case of mixture-order interactions. For example, one could consider higher-order interactions among PWO variables (Mee, 2020) and their possible interactions with the mixture proportions. The models presented here could be extended to generalized linear models, which has seen some attention in mixture experiments (Akay and Tez, 2007). Furthermore, other methods for identifying optimal mixture proportions and order may be of interest for exploration aside from Simulated Annealing. When the number of components m is large, the brute-force approach

essentially creates a branch for each possible order and then applies a maximization or minimization routine. In this case, it may be of interest to prune some of these branches using a branch and bound algorithm, which has seen some success in mixed-integer programming problems (Gupta and Ravindran, 1985; Wolsey, 2007).

There are several other approaches that may be taken to find efficient fixed-run OofA Mixture designs, which would extend the TA algorithm proposed in Chapter 4. First, the implementation of the TA algorithm is not necessarily limited to the $D-$ or $I-$optimality criteria. As long as the update to the design criterion is not computationally burdensome for a single-row exchange, the methods in this chapter still apply. Moreover, a theoretical framework for optimal OofA Mixture designs still needs to be established. Finally, we assumed that a list of candidate points is provided. It is also of interest to use an algorithm that does not rely on candidate points, such as a genetic algorithm (Pradubsri et al., 2019) or other metaheuristic algorithms (García-Ródenas et al., 2020).

There is much future work to be done with regard to graphical OofA experiments. It is of interest to explore other design criteria apart from $D-$ and $A-$optimality. In particular, criterion that emphasize prediction over parameter estimation (such as I-optimality) may be of interest in this problem. If we are only concerned about prediction over the set of all Hamiltonian paths on a graph, a suitable optimality criterion should seek to minimize the prediction variance over this particular set of runs. If there are heavy constraints, then the resulting set of Hamiltonian paths would be small, and predictions could be made on all possible runs. One can also consider the use of other surrogate models in place of the PWO model, such as the Component-Position (CP) model (Yang et al., 2021), or even a Gaussian Process model (Xiao and Xu, 2021). Another area for future development would be the extension of this work to directed graphs, as many practical applications only have directed edges (e.g. one-way roads). Furthermore, in Chapter 5, it was assumed that the graphical constraints were known prior to designing the experiment. If this is not the case, then the graphical model must first be estimated. There are several publications that explore learning the underlying graphical model based on a dataset; Lee et al. (2022) use a penalized likelihood approach to estimate parameters in ordinal graphical models, Lee and Xue (2018) also uses penalized likelihood to fit nonparametric finite mixtures of gaussian graphical mdoels, and Xue and Zou (2012) investigates rank-based estimation of high-dimensional gaussian graphs. More work would need to be done to develop the appropriate methodology for learning the graphical constraints in an OofA experiment.

Bibliography

Aidoo, R. P., E. O. Afoakwa, and K. Dewettinck (2014). Optimization of inulin and polydextrose mixtures as sucrose replacers during sugar-free chocolate manufacture–rheological, microstructure and physical quality characteristics. *Journal of Food Engineering 126*, 35–42.

Akay, K. U. and M. Tez (2007). Analyzing mixture experiments via generalized linear models. *International Journal of Pure and Applied Mathematics 36*(3), 373.

Al-Lazikani, B., U. Banerji, and P. Workman (2012). Combinatorial drug therapy for cancer in the post-genomic era. *Nature Biotechnology 30*(7), 679–692.

Barabási, A.-L. and R. Albert (1999). Emergence of scaling in random networks. *Science 286*(5439), 509–512.

Bertsimas, D. and J. Tsitsiklis (1993). Simulated annealing. *Statistical science 8*(1), 10–15.

Box, G. E. and N. R. Draper (1963). The choice of a second order rotatable design. *Biometrika*, 335–352.

Chandrasekaran, S. M., S. Bhartiya, and P. P. Wangikar (2006). Substrate specificity of lipases in alkoxycarbonylation reaction: Qsar model development and experimental validation. *Biotechnology and Bioengineering 94*(3), 554–564.

Chen, J., R. Mukerjee, and D. K. J. Lin (2020). Construction of optimal fractional order-of-addition designs via block designs. *Statistics & Probability Letters 161*, 108728.

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

Cook, R. D. and C. J. Nachtrheim (1980). A comparison of algorithms for constructing exact d-optimal designs. *Technometrics 22*(3), 315–324.

Cornell, J. A. (1990). *Experiments with Mixtures: Designs, Models and the Analysis of Mixture Data*. John Wiley & Sons.

Crosier, R. B. (1984). Mixture experiments: geometry and pseudocomponents. *Technometrics 26*(3), 209–216.

Crosier, R. B. (1986). The geometry of constrained mixture experiments. *Technometrics 28*(2), 95–102.

DeGroot, M. (1970). *Optimal Statistical Decisions*. McGraw Hill.

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

Fedorov, V. (1972). *Theory of Optimal Experiments. Probability and Statistics*. Academic Press, New York, New York.

Frieze, A. and M. Karoński (2016). *Introduction to Random Graphs*. Cambridge University Press.

García-Ródenas, R., J. C. García-García, J. López-Fidalgo, J. Á. Martín-Baos, and W. K. Wong (2020). A comparison of general-purpose optimization algorithms for finding optimal approximate experimental designs. *Computational Statistics & Data Analysis 144*, 106844.

Ghalanos, A. and S. Theussl (2012). Rsolnp: General non-linear optimization using augmented lagrange multiplier method. *R package version 1*.

Gilbert, E. N. (1959). Random graphs. *The Annals of Mathematical Statistics 30*(4), 1141–1144.

Goos, P., B. Jones, and U. Syafitri (2016). I-optimal design of mixture experiments. *Journal of the American Statistical Association 111*(514), 899–911.

Gupta, O. K. and A. Ravindran (1985). Branch and bound experiments in convex nonlinear integer programming. *Management science 31*(12), 1533–1546.

Hagberg, A., P. Swart, and D. S Chult (2008). Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

Hardin, R. and N. Sloane (1993). A new approach to the construction of optimal designs. *Journal of Statistical Planning and Inference 37*(3), 339–369.

Khalila, O. A., F. Harirchia, D. Kimc, S. Liua, P. Elvatic, A. Violic, and A. O. Heroa (2017). Model reduction in chemical reaction networks: A data-driven sparse-learning approach. *arXiv preprint arXiv:1712.06281*.

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

Laguna, M. and R. Marti (2005). Experimental testing of advanced scatter search designs for global optimization of multimodal functions. *Journal of Global Optimization 33*(2), 235–255.

Lee, K. H., Q. Chen, W. S. DeSarbo, and L. Xue (2022). Estimating finite mixtures of ordinal graphical models. *Psychometrika 87*(1), 83–106.

Lee, K. H. and L. Xue (2018). Nonparametric finite mixture of gaussian graphical models. *Technometrics 60*(4), 511–521.

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

Lyra, M., J. Paha, S. Paterlini, and P. Winker (2010). Optimization heuristics for determining internal rating grading scales. *Computational Statistics & Data Analysis 54*(11), 2693–2706.

Mee, R. W. (2020). Order-of-addition modeling. *Statistica Sinica 30*(3), 1543–1559.

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

Papadakis, M., M. Tsagris, M. Dimitriadis, S. Fafalios, I. Tsamardinos, M. Fasiolo, G. Borboudakis, J. Burkardt, C. Zou, K. Lakiotaki, and C. Chatzipantsiou. (2022). *Rfast: A Collection of Efficient and Extremely Fast R Functions*. R package version 2.0.6.

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

Piepel, G. F. (1983). Defining consistent constraint regions in mixture experiments. *Technometrics 25*(1), 97–101.

Pradubsri, W., B. Chomtee, and J. J. Borkowski (2019). Using a genetic algorithm to generate d-optimal designs for mixture-process variable experiments. *Quality and Reliability Engineering International 35*(8), 2657–2676.

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

Rusek, K., J. Suárez-Varela, A. Mestres, P. Barlet-Ros, and A. Cabellos-Aparicio (2019). Unveiling the potential of graph neural networks for network modeling and optimization in sdn. In *Proceedings of the 2019 ACM Symposium on SDN Research*, pp. 140–151.

Sljivic-Ivanovic, M. Z., I. D. Smiciklas, S. D. Dimovic, M. D. Jovic, and B. P. Dojcinovic (2015). Study of simultaneous radionuclide sorption by mixture design methodology. *Industrial & Engineering Chemistry Research 54*(44), 11212–11221.

Smith, K. (1918). 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.

Snee, R. D. and D. W. Marquardt (1974). Extreme vertices designs for linear mixture models. *Technometrics 16*(3), 399–408.

Suárez-Varela, J., S. Carol-Bosch, K. Rusek, P. Almasan, M. Arias, P. Barlet-Ros, and A. Cabellos-Aparicio (2019). Challenging the generalization capabilities of graph neural networks for network modeling. In *Proceedings of the ACM SIGCOMM 2019 Conference Posters and Demos*, pp. 114–115.

Van Nostrand, R. (1995). Design of experiments where the order of addition is important. In *ASA proceedings of the Section on Physical and Engineering Sciences*, pp. 155–160. American Statistical Association Alexandria, VA.

Voelkel, J. G. (2019). The design of order-of-addition experiments. *Journal of Quality Technology 51*(3), 230–241.

Voelkel, J. G. and K. P. Gallagher (2019). The design and analysis of order-of-addition experiments: An introduction and case study. *Quality Engineering 31*(4), 627–638.

Wald, A. (1943). On the efficient design of statistical investigations. *The Annals of Mathematical Statistics 14*(2), 134–140.

Wheeler, B. (2019). *AlgDesign: Algorithmic Experimental Design*. R package version 1.2.0.

Winker, P. (2000). *Optimization Heuristics in Econometrics: Application of Threshold Accepting*. Wiley.

Winker, P., J. Chen, and D. K. J. Lin (2020). The construction of optimal design for order-of-addition experiment via threshold accepting. In *Contemporary Experimental Design, Multivariate Analysis and Data Mining*, Chapter 6, pp. 93–109. Springer.

Wolsey, L. A. (2007). Mixed integer programming. *Wiley Encyclopedia of Computer Science and Engineering*, 1–10.

Xiao, Q. and H. Xu (2021). A mapping-based universal kriging model for order-of-addition experiments in drug combination studies. *Computational Statistics & Data Analysis 157*, 107155.

Xue, L. and H. Zou (2012). Regularized rank-based estimation of high-dimensional nonparanormal graphical models. *The Annals of Statistics 40*(5), 2541–2571.

Yang, J.-F., F. Sun, and H. Xu (2021). A component-position model, analysis and design for order-of-addition experiments. *Technometrics 63*(2), 212–224.

Zhao, Y., D. K. Lin, and M.-Q. Liu (2021). Designs for order-of-addition experiments. *Journal of Applied Statistics 48*(8), 1475–1495.

Zhao, Y., D. K. Lin, and M.-Q. Liu (2022). Optimal designs for order-of-addition experiments. *Computational Statistics & Data Analysis 165*, 107320.

Vita

Nicholas Rios

Education

PhD, Statistics *Expected August 2022*
The Pennsylvania State University
Dissertation Title: "Contributions to Mixture Experiments: Optimality when Order of Addition is Important"
Advisors: Dennis K.J. Lin, Lingzhou Xue

M.S., Statistics *December 2017*
Montclair State University
Master's Thesis Title: "Prediction Intervals for Functional Data"
https://digitalcommons.montclair.edu/etd/1
Advisor: Andrada Ivanescu

B.S., Statistics, Summa Cum Laude *December 2014*
University of Delaware
Minors: Computer Science, Economics, Mathematics

Publications

Rios, Nicholas, Winker, Peter, and Lin, Dennis K.J. "TA Algorithms for D-Optimal OofA Mixture Designs." (2021). *Computational Statistics & Data Analysis. 168*, 107411.

Rios, Nicholas, and Lin, Dennis K.J. "Order-of-Addition Mixture Experiments." (2021). *Journal of Quality Technology*: 1-10.

Rios, Nicholas, and Lin, Dennis K.J. "Graphical Methods for Order-of-Addition Experiments." (2022). Submitted to JASA.

Quinlan, Kevin, Rios, Nicholas, Kravvaris, Konstantinos, and Quaglioni, Sofia. "Parallel Partial Cokriging with Sequential Design of Experiments for $n - \alpha$ Scattering." (2021). *Journal of Statistical Planning and Inference.* Submitted for Review.

Mahadevan, Seshasayee, Rios, Nicholas, Abbey J. Kollar, Rachel Stofanak, Katherine Maloney, Kayley.E Waltz, Chinmayee Rane, Sandeep Endluri and Phillip E. Savage. "Statistical Models for Predicting Oil Composition from Hydrothermal Liquefaction of Biomass." (2021). To be presented at 2021 AIChE Annual Meeting.