Computational Statistics and Data Analysis 168 (2022) 107411

Contents lists available at ScienceDirect

Computational Statistics and Data Analysis

www.elsevier.com/locate/csda

TA algorithms for D-optimal OofA Mixture designs

Nicholas Rios [a,*], Peter Winker [b], Dennis K.J. Lin [c]

[a] Department of Statistics, The Pennsylvania State University, State College, PA, 16802, United States of America
[b] Department of Economics, University of Giessen, Licher Strasse 64, 35394, Giessen, Germany
[c] Department of Statistics, Purdue University, West Lafayette, IN, 47907, United States of America

ARTICLE INFO

Article history:
Received 24 May 2021
Received in revised form 5 November 2021
Accepted 3 December 2021
Available online 9 December 2021

Keywords:
D-optimal designs
Design of experiments
Pairwise ordering model
Threshold accepting algorithms

ABSTRACT

In a mixture experiment, m components are mixed to produce a response. The total amount of the mixture is a constant. This classical experiment has been studied for a long time, but little attention has been given to the addition order of the components. In an Order-of-Addition (OofA) Mixture experiment, the response depends on both the mixture proportions of components and their order of addition. The overall goal of the OofA Mixture experiment is to identify the addition order and mixture proportions that produce an optimal response. Methodology for constructing full OofA Mixture designs is discussed, but the size of these full designs increases rapidly as m increases. A Threshold Accepting (TA) algorithm is used to find a subset of n rows of the full OofA Mixture design that maximize the D-optimality criterion, reducing the number of required runs. Neighborhood structures are proposed for OofA simplex lattice and general mixture designs. The TA algorithm is compared with the well-known Fedorov algorithm, and recommendations for the use of this algorithm are provided.

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, this means finding the values of the mixture proportions that maximize a response, minimize a response, or have the response match a pre-existing target value.

In an Order-of-Addition experiment, the response depends on the order in which m components are added to a system. The Order-of-Addition problem surfaces in many practical applications. In the pharmaceutical industry, Rajaonarivony et al. (1993), showed that the size of nanoparticles formed was found to depend on the order of addition of poly-L-lysine and calcium to sodium-alginate solution. Additionally, in Chandrasekaran et al. (2006), the efficiency of synthesis of carbonate products depended on the order of addition of alcohols. Another example is the field of combinatorial drug therapy, where Ding et al. (2015) demonstrate that both the ratio and order of three drugs have an impact on the treatment of oral cancer. The OofA experiment has been well-explored by Lin and Peng (2019) and Peng et al. (2019).

In an Order-of-Addition Mixture experiment (OofA Mixture), researchers are concerned with both the addition order of the components and their mixture proportions. Assuming that the m components are sequentially mixed, there are m!

* Corresponding author.
E-mail address: nar203@psu.edu (N. Rios).

https://doi.org/10.1016/j.csda.2021.107411
0167-9473/© 2021 Elsevier B.V. All rights reserved.

possible orderings. 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. See Rios and Lin (2021), where an algorithm is proposed for creating such a design in the case where there are no constraints on the mixture proportions.

The focus of this paper is on designing experiments for the OofA Mixture experiment in the case where funds are limited, and researchers can only afford a small number of runs. This will be accomplished by using the Threshold Accepting (TA) algorithm to select a small number of runs from a larger pool of possible runs according to the popular D-optimality criterion. There are many possible criteria for selecting an optimal design (e.g. A-,I-), and the TA algorithm proposed in this paper may be adapted to different criteria without changing the overall algorithm. The Threshold Accepting heuristic originally proposed by Dueck and Scheuer (1990) has been widely used, specifically for addressing experimental design problems. The first of these applications focused on the (approximate) evaluation of the star discrepancy (Winker and Fang, 1997). Further contributions focused on the construction of low discrepancy and optimal designs on a simplex (e.g. Fang et al. 2000, Fang et al. 2006), optimal designs on flexible regions (Lin et al., 2010) and robust designs (Winker and Lin, 2011).

1.1. Order of addition experiments

In an Order-of-Addition (OofA) experiment, a measurable response y depends on the order in which m components are added to a system or compound. Most of the existing work on the OofA problem is on the Pairwise Ordering (PWO) model, which was introduced by Van Nostrand (1995). It was officially called the Pairwise 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 $1 \le j < k \le m$. Let jk denote the pair (j, k). The PWO factor for all $jk \in \mathcal{P}$ is defined as

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

So if $\mathbf{a} = (3, 1, 2)$ then $z_{12}(\mathbf{a}) = 1, z_{13}(\mathbf{a}) = -1, z_{23}(\mathbf{a}) = -1$. The PWO factors must obey the transitive property, i.e., 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 $z_{12}(\mathbf{a}) = 1, z_{13}(\mathbf{a}) = -1, z_{23}(\mathbf{a}) = 1$. Let $\tau(\mathbf{a})$ be the expected response given permutation $\mathbf{a}$. The PWO model is

$$\tau(\mathbf{a}) = \delta_0 + \sum_{jk \in \mathcal{P}} z_{jk}(\mathbf{a})\delta_{jk}\,. \tag{2}$$

In Equation (2), δ_{jk} is the effect of the order of components j and k on the response. The parameter estimates from the PWO model (2) may be used to help identify an optimal order of addition. Lin and Peng (2019) discuss how topological sorting methods may be used to finding an optimal order given the output of a PWO model. There has also been a large amount of research conducted on the optimality of PWO designs. Peng et al. (2019) proved that the full PWO design (with all $m!$ runs) is optimal for any criterion that is concave and signed permutation invariant, which covers the $D-$, $A-$, $E-$, and $M.S.-$ criteria. Peng et al. (2019) went a step further, and provided a systematic method for constructing fractional PWO designs that have the same moment matrix as the full design, and are therefore optimal. TA algorithms were demonstrated to be useful in constructing OofA designs in Winker et al. (2020), so it is natural to consider this method in the extension to OofA Mixture experiments. For more papers on OofA designs, see also Chen et al. (2020) and Zhao et al. (2021).

1.2. Mixture experiments

In a mixture experiment, m components are mixed to produce a (continuous) response y. The total amount of the mixture is fixed. Let $x_1, \ldots, x_m$ represent the proportions of the m components in a mixture experiment, with $0 \le x_i \le 1$ 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 two classical mixture designs: the simplex-lattice design, and the extreme vertices design. Many other classical designs and models for mixture experiments are available (Cornell, 1990), and our methods are not limited to these designs.

Simplex-Lattice Design. A classical approach is the $\{m, l\}$ simplex lattice design. Here, m is the number of mixture components, and l is the degree of the largest polynomial model that should be fit to the response surface. This design is formed by listing all possible combinations of the points $x_i = 0, \frac{1}{l}, \frac{2}{l}, \ldots 1$ that are inside the simplex $\mathcal{S}$.

Extreme Vertices Design. In practice, it is likely that constraints will be placed on the mixture proportions. Suppose that there are single-component constraints $0 \le L_i \le x_i \le U_i \le 1$ for each $i = 1, 2, \ldots, m$. In this case, the experimental region is a polyhedron that lies within the simplex. The extreme vertices design chooses the vertices of the bounded experimental region as design points, in addition to center points on the faces and interior of the region. Several algorithms can construct this design, such as Snee and Marquardt (1974). It is rare in a practical application to see a degree higher than 2 used to model the response surface. Therefore, we focus on the second-order polynomial to model the response surface:

$$\eta = \sum_{i=1}^{m} \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j \,, \tag{3}$$

where η represents the response surface, x_i is the value taken by the i^{th} mixture component, β_i is the effect of a single blend of component i, while β_{ij} is the effect of a mixed blend of components i and j on the response. The model has no intercept term due to the constraint that $\sum_i x_i = 1$. Model (3) can be extended to a mixture-process experiment, which occurs when variables other than the mixture proportions (called process variables) affect the response. Examples of process variables are temperature, time, or pressure. A common approach to designing such an experiment is to create a 2^k factorial design for the process variables, and then run a mixture experiment at each combination of the process variables (Cornell, 1990). This allows researchers to determine the effect of interactions between the mixture proportions and process variables.

In reality, it can be impractical to use all runs from a mixture design. Instead of randomly selecting a subset of these runs, one can choose a subset using an optimality criterion. In a mixture model $y = X\beta + \epsilon$, where the errors are independent with constant variance, the variance of the model parameters is proportional to $(X^T X)^{-1}$. All optimality criteria choose a matrix X that "minimizes" $(X^T X)^{-1}$ in some way. The D-optimality criterion tries to minimize $\det((X^T X)^{-1})$, which minimizes the volume of the confidence ellipsoid about the coefficients β. The I-optimality criterion tries to minimize $\int_{x\in\mathcal{S}} f(x)(X^T X)^{-1} f(x)^T \, dx$, i.e., the average prediction variance over the experimental region under a mixture model f (Goos et al., 2016). We focus on the D-optimality criterion as an example because of its popularity in the OofA literature and its ease of implementation. It is possible to apply the methods in our paper to any optimality criterion that may be quickly updated.

1.3. *OofA Mixture experiments*

In an OofA Mixture experiment, the response depends on both the mixture proportions of m components and their addition order into the mixture. Researchers often have three main goals of interest in this experiment: (1) perform statistical inference on the mixture components; (2) perform statistical inference on the addition order of the components; (3) identify the addition order and proportions of the m components that produce an optimal response. In this paper, we first identify a method for enumerating all sensible experimental runs for an arbitrary OofA Mixture experiment. However, these "full" designs are often quite large. We hope to reduce the size of these OofA Mixture designs, while still being able to achieve the previously mentioned goals. For more details on the OofA Mixture problem, see Section 2.

There are several studies which conclude that the addition order of mixture components may have an impact on the response. Voelkel and Gallagher (2019) studied the effect of the order of addition of mixture components on the viscocity of automotive paint coatings. Although this is indeed a mixture experiment, only the order of addition was studied in this case, as the mixture proportions were held constant. Sljivic-Ivanovic et al. (2015) studied how applying $m = 3$ sorbents in a mixture impacted the removal of ions from aqueous solutions. They concluded that the order of addition was indeed significant. It should also be noted that they conducted separate experiments for the mixture proportions and the order of addition; they used an extreme vertices design to examine the mixture proportions, and they ran all $m! = 6$ orderings with fixed mixture proportions to study the order of addition.

1.4. *Outline of paper*

In Section 2, we describe how to construct full designs for the OofA Mixture experiment, and formally define the problem of selecting a design matrix of n runs from a full OofA Mixture design matrix. Furthermore, we describe how to use the Threshold Accepting (TA) algorithm to find D-optimal designs. Section 3 shows results regarding the implementation and performance of the TA algorithm and recommendations for certain algorithm parameters. Section 4 provides two examples that compare the TA algorithm to the well-known Fedorov exchange algorithm. Section 5 concludes the paper.

2. Methods

In this section, we (1) describe how to construct full OofA Mixture designs that include all feasible combinations of a set of mixture proportions and addition orders, (2) formally state the problem of finding a D-optimal subset of these full designs, and (3) describe Fedorov and TA algorithms for solving such a problem.

Table 1
An OofA simplex lattice design for $m = 3, l = 2$.

x_1	x_2	x_3	z_{12}	z_{13}	z_{23}
1.0	0.0	0.0	0.0	0.0	0.0
0.0	1.0	0.0	0.0	0.0	0.0
0.0	0.0	1.0	0.0	0.0	0.0
0.5	0.5	0.0	1.0	0.0	0.0
0.5	0.5	0.0	-1.0	0.0	0.0
0.5	0.0	0.5	0.0	1.0	0.0
0.5	0.0	0.5	0.0	-1.0	0.0
0.0	0.5	0.5	0.0	0.0	1.0
0.0	0.5	0.5	0.0	0.0	-1.0

2.1. *Full designs and models for OofA Mixture*

In an Order-of-Addition (OofA) mixture experiment, we assume that the response depends on both the mixture proportions $\mathbf{x} = (x_1, \ldots, x_m)$ of each component, and the order of addition of these m components. There are $m!$ possible orderings of the m components. Let $\mathbf{a}$ be a permutation (ordering) of $(1, 2, \ldots, m)$ and $\mathbf{x} = (x_1, \ldots, x_m)$. In general, suppose that

$$y = f(\mathbf{x}, \mathbf{a}) + \epsilon\,. \tag{4}$$

The goal of a mixture experiment is to find the optimal mixture proportions and ordering, i.e., to find the mixture proportions and addition order that maximize the response or bring it close to a pre-determined target value T. Let $\mathcal{A}$ be the set of all permutations of $(1, 2, \ldots, m)$. Let $\mathcal{S}^* = \{(x_1, \ldots, x_m) : L_i \le x_i \le U_i,\ \ \sum_i x_i = 1\}$. Then, if the goal is maximization of the response, we may write this as

$$(\mathbf{x}^*, \mathbf{a}^*) = \underset{\mathbf{x},\mathbf{a}}{\arg\max}\, f(\mathbf{x}, \mathbf{a}) \quad \text{subject to} \quad \mathbf{a} \in \mathcal{A} \quad \text{and} \quad \mathbf{x} \in \mathcal{S}^*\,. \tag{5}$$

To represent the order of addition of the m mixture components, we define these Mixture Pairwise Ordering (MPWO) variables:

$$z_{jk}(\mathbf{x}, \mathbf{a}) = \begin{cases} 1 & x_j, x_k \neq 0 \ \text{ and } j \text{ is before } k \text{ in } \mathbf{a}\,; \\ 0 & x_j = 0 \text{ or } x_k = 0 \text{ or both;} \\ -1 & x_j, x_k \neq 0 \ \text{ and } j \text{ is after } k \text{ in } \mathbf{a}. \end{cases} \tag{6}$$

For example, if the order of addition of $m = 3$ components is $\mathbf{a} = (3, 1, 2)$, but the mixture proportions are $\mathbf{x} = (0.2, 0, 0.8)$, then $z_{13}(\mathbf{x}, \mathbf{a}) = -1$ (since components 1 and 3 were included in the mixture, and component 1 was added after 3), but $z_{12}(\mathbf{x}, \mathbf{a}) = 0$ because component 2 was not included in the mixture. Using these MPWO variables, it is possible to construct a full design matrix D_{full} using Algorithm 1.

Algorithm 1: Generate full OofA Mixture design matrix.

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

Algorithm 1 takes an arbitrary design for the m mixture components, X^*. Each row of X^* is repeated $k!$ times, where k is the number of nonzero components in the row. Then, orderings are assigned to these nonzero components. This creates a “full” design in the sense that every possible ordering of the nonzero components is checked. Table 1 shows an example of a simplex lattice design matrix D for $m = 3$ components with degree $l = 2$ for the OofA Mixture problem.
In an OofA Mixture experiment, the response y has the following form:

$$y = \eta(\mathbf{x}) + g(\mathbf{x}, \mathbf{a}) + \epsilon\,. \tag{7}$$

Here, $\eta(\mathbf{x})$ is the “pure mixture” model, and it is typically a second-order polynomial mixture model of the m components, i.e.,

Table 2
Run size (N) for full $\{m, l\}$ 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

$$\eta(\mathbf{x}) = \sum_{i=1}^{m} \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j \,. \tag{8}$$

There is some flexibility in the choice of g. One can choose g so that there are no mixture-order interactions. In this case, the full OofA Mixture model can be written as

$$\eta(\mathbf{x}) = \sum_{i=1}^{m} \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j, \text{ and } g(\mathbf{x}, \mathbf{a}) = \sum_{j<k} \delta_{jk} z_{jk}(\mathbf{x}, \mathbf{a}) \,, \tag{9}$$

for all possible pairs $j < k$. It is also possible to allow for said mixture-order interactions, as in the following models:

$$g(\mathbf{x}, \mathbf{a}) = \sum_{i} \sum_{k<l} \gamma_{kl}^{i} x_i z_{kl}(\mathbf{x}, \mathbf{a}), \text{ or,} \tag{10}$$

$$g(\mathbf{x}, \mathbf{a}) = \sum_{k<l} \delta_{kl} z_{kl}(\mathbf{x}, \mathbf{a}) + \sum_{i} \sum_{k<l, i=k,l} \gamma_{kl}^{i} x_i z_{kl}(\mathbf{x}, \mathbf{a}) \,. \tag{11}$$

Models (10) and (11) are based on the mixture-process variable models (Cornell, 1990). Model (10) does not include any main pairwise order effects, but Model (11) does. Note that Model (11) assumes that all interaction effects $\gamma_{kl}^{i} = 0$ if $i \neq k$ and $i \neq l$.

2.2. Problem statement

In general, existing OofA Mixture designs have too many runs. Table 2 shows the run sizes for OofA Simplex-Lattice Designs for various numbers of components m and degrees l. In Table 2, the run sizes increase rapidly as m and l increase.

Given an OofA Mixture 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. Let $\mathcal{D}_n$ be the collection of all matrices with exactly 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, M^* is D-optimal if

$$M^* = \operatorname*{arg\,min}_{M_n \in \mathcal{M}_n} \det((M_n^T M_n)^{-1}) = \operatorname*{arg\,max}_{M_n \in \mathcal{M}_n} \det(M_n^T M_n) \,. \tag{12}$$

Once M^* is found, we know the corresponding optimal design matrix D^*. Throughout this paper, we will work mainly with the model matrices $M_n \in \mathcal{M}_n$ using a given model f of the general form (7). Once an optimal design is identified, then the corresponding OofA Mixture experiment can be executed. Using the data from this experiment, an OofA Mixture model of the form (7) can be fit. This model can be used to estimate the response for new mixture proportions and addition orders, which can be used to identify the optimal mixture proportions and order of addition.

2.3. TA algorithms for OofA Mixture designs

In this section, we describe how to use Threshold Accepting (TA) algorithms for selecting D-optimal OofA Mixture designs of a fixed size. 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. The pseudo-code for a general Threshold Accepting (TA) algorithm for finding a D-optimal design is given in Algorithm 2.

Algorithm 2: Pseudo code for generic TA algorithm for finding D-optimal design.

```
Input: A sequence $\{t_r : r = 1, 2, \dots\}$ 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 $D_0$.
```

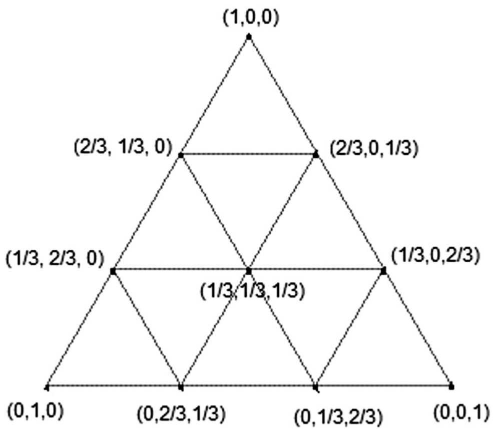

Fig. 1. The points in a $\{3, 3\}$ SLD. Neighboring points have edges between them.

We denote the D-efficiency of a design D as $de(D)$. In step 1, a design D_0 is randomly selected from a set of candidate points. Then, a new design D_1 is chosen that is in the neighborhood of D_0, i.e., the new design is not very "different" from the starting design. In Algorithm 2, the sequence $\{t_r : r = 1, 2, \dots\}$ tends toward 0 as r increases. This ensures that a new design is only selected if it has a higher D-efficiency than that of the previous design, or if it has a D-efficiency that is slightly worse than that of the previous design (where slightly is defined by the choice of t_r). This helps the algorithm avoid premature convergence. When t_r is large, it is possible to choose some designs that are slightly less ideal in terms of D-efficiency. Then, as r increases, t_r decreases, allowing the algorithm to converge.

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

Definition 2. Let D and D' be two $\{m, l\}$ 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, denote this as $D' \in \mathcal{N}_k(D)$. The definition of pairwise adjacent lattice neighbors arises from the fact that the mixture points in an $\{m, l\}$ SLD are equally spaced, and the Euclidean distance between any two adjacent mixture points in such a design is $\frac{\sqrt{2}}{l}$. Hence, it is natural to conclude that two mixture points $\mathbf{x}, \mathbf{x}'$ are "neighbors" if the distance between them is $\frac{\sqrt{2}}{l}$. To illustrate this, Fig. 1 shows the points in a $\{3, 3\}$ SLD that are equally spaced with a distance $\frac{\sqrt{2}}{3}$. In Fig. 1, neighboring points have edges between them.

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, note 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 is useful 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 $\mathbf{d}'$ are neighbors 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.

Once a neighborhood is defined, it remains to discuss how to generate the threshold sequence $\{t_r : r = 1, 2, 3, \ldots\}$ An empirical procedure similar to that of Winker et al. (2020) is used to generate the threshold sequence. This procedure is summarized in Algorithm 3.

Algorithm 3: 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 $t = (0, \ldots, 0)$ of length $2n_i$.
for $r = 1, 2, \ldots, 2n_i$ 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. $t_r = |de(D_0) - de(D_1)|$
end
5. Sort $t$ in descending order.
6. Use the lower 50% of $t$ as the threshold sequence $\{t_r : r = 1, \ldots, n_i\}$ .
Output The threshold sequence $\{t_r : r = 1, \ldots, n_i\}$
```

Algorithm 3 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 an OofA Mixture 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 D-efficiency is stored in the list (steps 2-4). 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 5-6).

Algorithm 4: 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 3 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 $de(D_0), de(D_1)$.
    11. if $de(D_1) > de(D_0) - t_r$ then
        Update $D_0 = D_1, d_0 = d_1, C_0 = C_1$.
    end
end
Output $D_0$ and $de(D_0)$.
```

Algorithm 4 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 4 returns a design D_0 that is D-efficient, and it also displays the D-efficiency of D_0. Step 1 of Algorithm 4 calls Algorithm 3 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 4, 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 4 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

$$\Delta(\mathbf{x},\mathbf{y}) = 1 + v(\mathbf{y}) - v(\mathbf{x}) + v(\mathbf{x},\mathbf{y})^2 - v(\mathbf{y})v(\mathbf{x})\,, \tag{13}$$

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:

$$C_1 = C_0 - C_0 F_1 (I_2 + F_2^T C_0 F_1)^{-1} F_2^T C_0\,. \tag{14}$$

In equation (14), $C_1 = (M^T M)^{-1}$ is the inverse of the moment matrix after the exchange, M is the model matrix after the exchange, $C_0 = M_0^T M_0$, $F_1 = [f(\mathbf{y}), -f(\mathbf{x})]$, and $F_2 = [f(\mathbf{y}), f(\mathbf{x})]$. 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 $de(D_1) > de(D_0) - t_r$ then the exchange is confirmed, otherwise the loop continues.

To implement Algorithms 3 and 4, 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 4 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 4 is compared with an updated version of Fedorov's exchange algorithm (Cook and Nachtrheim, 1980) that is outlined in Algorithm 5. Lines 1 to 3 of Algorithm 5 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 5: 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 $\Delta$.
        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 $\Delta$.
        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 $D_0$
```

Table 3
IQR (and standard deviation) for relative D-efficiencies of designs chosen by TA algorithm.

m	l	Number of iterations		
		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)

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

$$\delta = \frac{|M_t^T M_t| - |M_{t-1}^T M_{t-1}|}{|M_{t-1}^T M_{t-1}|}. \tag{15}$$

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.

3. Impact of algorithm parameters

This section focuses on the implementation of Algorithm 4 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. By double-point design, we mean a design with $n = 1 + 2p$ runs, where p is the number of model parameters. To put this into context, the smallest number of runs possible is 1+p, with the extra run used to estimate the error variance. To measure the performance of the TA algorithm, we rely on the relative D-efficiency of the n-run 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

$$\text{Relative D-efficiency} = 100\%\left(\frac{\frac{1}{n}|M^T M|^{1/p}}{\frac{1}{N}|M_{full}^T M_{full}|^{1/p}}\right). \tag{16}$$

In general, as the number of exchanges k increases, the relative D-efficiency decreases. For this reason, the number of exchanges was set to $k = 1$. For simplicity, we use the additive form of the OofA Mixture model (9).

We first examine the convergence of the TA algorithm as the number of iterations increases. Table 3 shows 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 $l \le \min(m, 5)$. In this table, $n = 1 + 2p$, where $p = m + 2\binom{m}{2}$ is the number of parameters in model (9). In all but two cases (noted below), the value of n did not greatly effect convergence of the algorithm.

In Table 3, in most cases, the IQR and standard deviation of the relative D-efficiencies decrease as the number of iterations increases. 100000 iterations appear 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 $l = 3$, 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 $m = 8, l = 3$, 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).

Fig. 2 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 increases, the center of the distribution typically increases.

To see how the relative D-efficiency changes as the target run size n increases (as a percentage of N), we examine the median relative D-efficiency for all combinations of m, l with $l \le m$ for $m = 4, 6, 8$. Table 4 shows these median relative D-efficiencies for $n = 0.3N, 0.4N, 0.5N, 0.6N, 0.7N, 0.8N$, and $0.9N$ for 10 random seeds and 100000 iterations. Fig. 3 shows plots for $m = 6, l = 3$ (panel a) and $m = 6, l = 4$ (panel b).

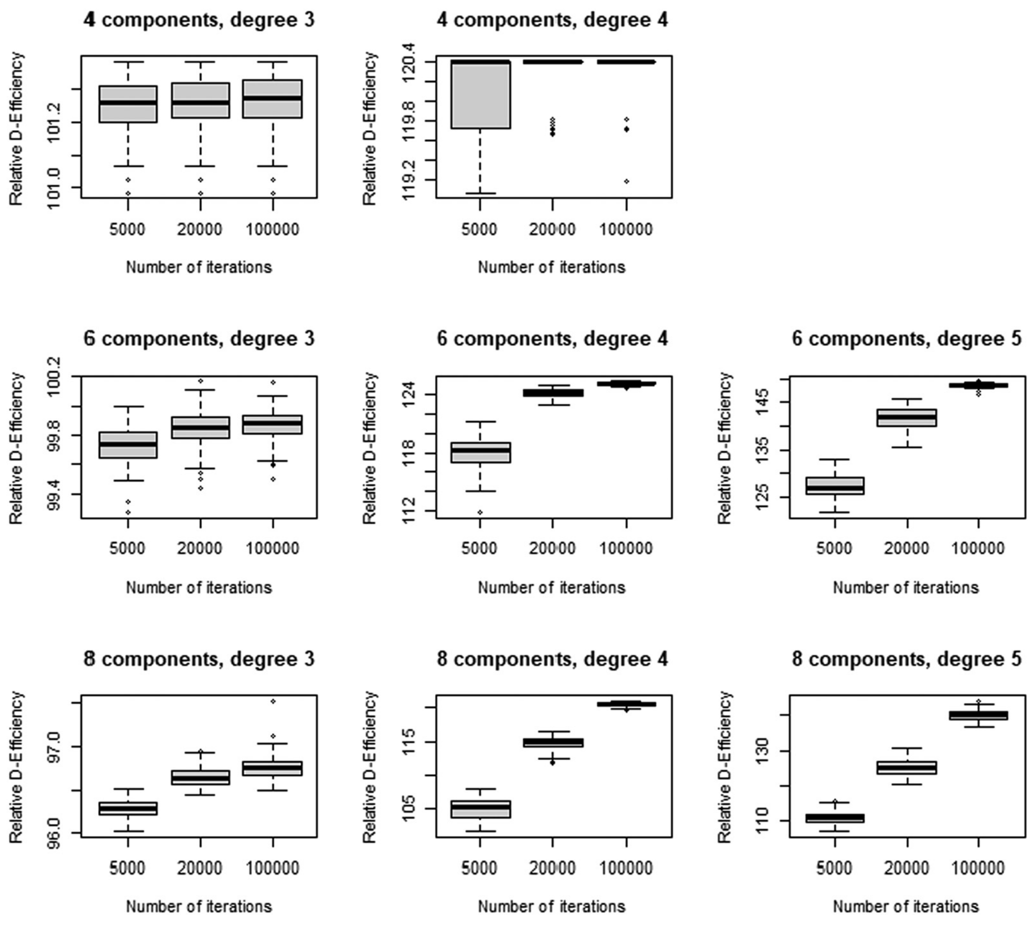

Fig. 2. Relative D-efficiency comparison for various $\{m, l\}$ OofA SLD designs at varying number of iterations of the TA algorithm.

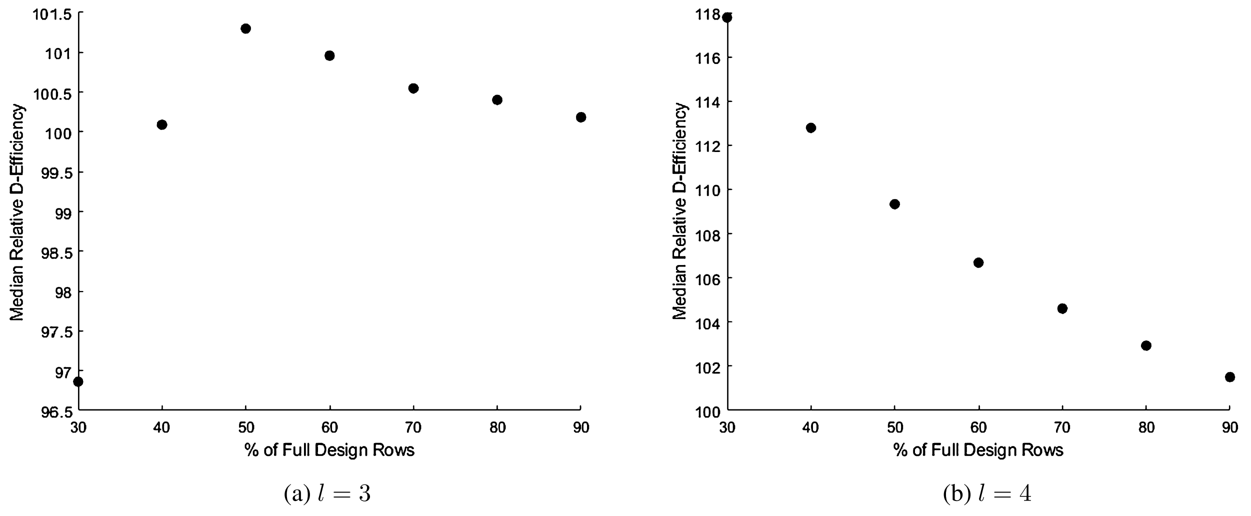

Fig. 3. Median relative D-efficiency of TA algorithm, $m = 6, l = 3, 4$.

The x-axes in Fig. 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, 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

Table 4
Median relative D-efficiencies for TA algorithm.

m	l	n						
		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 $n < p + 1$.

Table 5
Relative D-efficiency comparison for varying SLD degrees (100000 iterations).

m	l	n	100(n/N)%	Min	Q_{25}	Q_{50}	Q_{75}	Max
4	3	33	63.46	100.98	101.21	101.27	101.33	101.38
	4	33	24.26	119.19	120.39	120.39	120.39	120.39
6	3	73	39.24	99.49	99.81	99.88	99.93	100.16
	4	73	8.95	124.67	125.05	125.14	125.22	125.4
	5	73	2.43	146.74	148.52	148.71	148.89	149.57
8	3	129	28.29	96.49	96.66	96.74	96.83	97.52
	4	129	4.50	119.71	120.32	120.52	120.64	121.06
	5	129	0.82	136.74	138.98	140.1	140.92	144.12

size of the full design. In the other cases, the median relative D-efficiency decreases as n increases. In these cases, a smaller design is preferred. In Table 4, there are several cases where relative D-efficiencies are above 100%. 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. This will happen again in the examples in Section 4.

It is also of interest to see 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 results are shown in Table 5.

Table 5 shows the five-number summary of the relative D-efficiencies of 100 designs generated from the TA algorithm. More summary statistics may be found in Appendix B. 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. Comparison with exchange algorithm

It is of interest to compare the TA algorithm (Algorithm 4) 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 5). Algorithms 4 and 5 were implemented in Matlab. As defined in Section 3, we use the relative D-efficiency to compare designs. In Section 4.1, we show an example comparing these algorithms for a simplex-lattice design. In Section 4.2, these algorithms are compared in an example with single-component constraints and an extreme vertices design.

4.1. OofA simplex-lattice design

Both algorithms were run for $m = 4$ components, degree $l = 3$, and target sample size $n = 30$ ($N = 52$) for the additive OofA Mixture model (9). This example can be performed for a third-order pure mixture model as well, though we use the second-order model due to its popularity and simplicity. The Fedorov algorithm converges in 3 iterations. The TA algorithm was run for 100,000 iterations. The design chosen by the Fedorov Algorithm 5 has a relative D-efficiency of 101.6243% to the full design. The same initial points were used for each algorithm. Since the TA Algorithm 4 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 Fig. 4. Quantiles of the 1000 relative D-efficiencies are shown in Table 6.

In this case, the empirical distribution appears to be left-skewed. Table 6 shows that well over 50% of the designs found using the TA algorithm have a relative D-efficiency at least as good as the Fedorov algorithm. In fact, 71% of the designs generated by the TA algorithm have 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

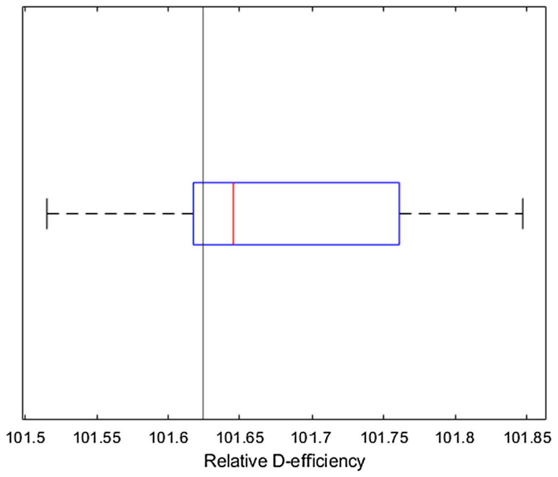

Fig. 4. Relative D-efficiencies of TA designs for 1000 random seeds. The vertical line represents the relative efficiency of the design obtained by the Fedorov algorithm.

Table 6
Quantiles of the relative D-efficiencies of 1000 designs generated by Algorithm 4.

$Q_{0.10}$	$Q_{0.25}$	Median	$Q_{0.75}$	$Q_{0.90}$
101.5922%	101.6176%	101.6457%	101.7608%	101.8471%

Table 7
Quantiles of the relative D-efficiencies of 1000 OofA EVDs generated by Algorithm 4.

$Q_{0.10}$	$Q_{0.25}$	Median	$Q_{0.75}$	$Q_{0.90}$
120.8555%	121.0650%	121.2949%	121.4874%	121.6593%

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 TA algorithm has an average runtime of 0.4675 seconds over all 1000 designs, while the Federov algorithm took 0.0190 seconds to converge. The design selected by the Federov algorithm, as well as the TA design with the highest relative D-efficiency, can be found in Appendix B.

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 design from Cornell (1990) has 15 runs, 8 of which are vertex points on the polyhedron, 6 of which are face centroids, and one overall centroid. See the Appendix 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 a D-optimal design with $n = N/6 = 60$ runs for Model (11), which includes both main effects and mixture-order interactions. As in Section 4.1, 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 chooses a design with a relative D-efficiency of 120.5293% (Fig. 5).

Based on Table 7, we can see that over 90% of the designs generated by the TA algorithm (roughly 98%) have higher relative D-efficiencies than the design generated by the Fedorov algorithm. The TA algorithm has an average runtime of 0.5261s over all 1000 designs, and the Federov algorithm has a runtime of 0.1479s. It is clear that in this case, the TA algorithm is preferred. This example also illustrates that (1) we can search for D-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. The design selected by the Federov algorithm, as well as the TA design with the highest relative D-efficiency, can be found in Appendix B.

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 have higher D-efficiencies than those found by the Federov algorithm. The first example shows a classical problem where the design space of the mixture proportions was

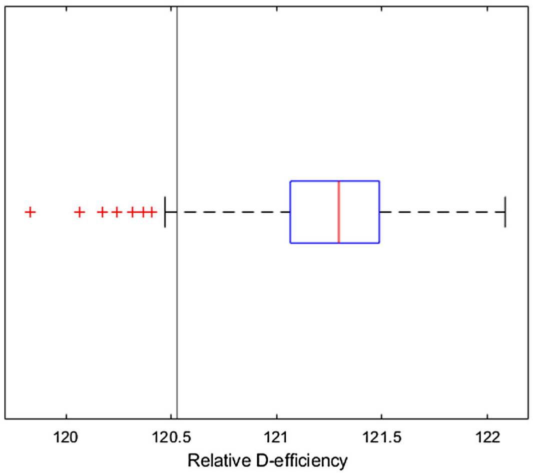

Fig. 5. Relative D-efficiencies of TA OofA EVDs for 1000 random seeds. The vertical line represents the relative efficiency of the design obtained by the Fedorov algorithm.

the $(m-1)$-dimensional simplex, while the second example deals 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.

5. Conclusion

To the best of our knowledge, this is the first paper that addresses 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 paper 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 paper, the TA algorithm was used to search for OofA Mixture designs of a fixed run size according to the D-optimality criterion. In Sections 3 and 4, we find 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 4, we give 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 paper can be used in many practical situations. The general definition of a neighborhood proposed in Section 2.3 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 paper can be used to search for D-optimal designs. This makes the method appealing for cases with single-component constraints, as shown in Section 4.2. Additionally, the TA algorithm can be applied to more design-specific neighborhoods, such as the simplex-lattice neighborhood constructed in this paper.

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 3, 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.

More work needs to be done with regards to finding optimal OofA Mixture designs. This paper only examined the D-optimality criterion. There are many other optimality criteria that would be of interest to study. Goos et al. (2016) pointed out that I-optimal designs are of importance in mixture experiments, as they minimize the average prediction variance over the experimental region, and the goal of a mixture experiment is to optimize the predicted model over this region. Moreover, a theoretical framework for optimal OofA Mixture designs still needs to be established. Finally, this paper 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). It would also be interesting to use OofA Mixture models to analyze an experimental dataset to see if mixture-order interactions are significant. Currently, most researchers do not examine this because the resulting design would be too large. Hopefully, the methods reviewed here will enable future research in this area.

Funding

Dennis Lin is supported by the U.S. National Science Foundation, via Grant #DMS-18102925.

Appendix. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.csda.2021.107411.

References

Chandrasekaran, S.M., Bhartiya, S., Wangikar, P.P., 2006. Substrate specificity of lipases in alkoxycarbonylation reaction: Qsar model development and experimental validation. Biotechnol. Bioeng. 94 (3), 554–564.
Chen, J., Mukerjee, R., Lin, D.K.J., 2020. Construction of optimal fractional order-of-addition designs via block designs. Stat. Probab. Lett. 161, 108728.
Cook, R.D., Nachtrheim, C.J., 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.
Ding, X., Matsuo, K., Xu, L., Yang, J., Zheng, L., 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.
Dueck, G., Scheuer, T., 1990. Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing. J. Comput. Phys. 90 (1), 161–175.
Fang, K.-T., Lin, D.K., Winker, P., Zhang, Y., 2000. Uniform design: theory and application. Technometrics 42 (3), 237–248.
Fang, K.-T., Maringer, D., Tang, Y., Winker, P., 2006. Lower bounds and stochastic optimization algorithms for uniform designs with three or four levels. Math. Comput. 75 (254), 859–878.
García-Ródenas, R., García-García, J.C., López-Fidalgo, J., Martín-Baos, J.Á., Wong, W.K., 2020. A comparison of general-purpose optimization algorithms for finding optimal approximate experimental designs. Comput. Stat. Data Anal. 144, 106844.
Goos, P., Jones, B., Syafitri, U., 2016. I-optimal design of mixture experiments. J. Am. Stat. Assoc. 111 (514), 899–911.
Lin, D.K.J., Peng, J., 2019. Order-of-addition experiments: a review and some new thoughts (with discussion). Qual. Eng. 31 (1), 49–59.
Lin, D.K.J., Sharpe, C., Winker, P., 2010. Optimized u-type designs on flexible regions. Comput. Stat. Data Anal. 54 (6), 1505–1515.
Lyra, M., Paha, J., Paterlini, S., Winker, P., 2010. Optimization heuristics for determining internal rating grading scales. Comput. Stat. Data Anal. 54 (11), 2693–2706.
Peng, J., Mukerjee, R., Lin, D.K.J., 2019. Design of order-of-addition experiments. Biometrika 106 (3), 683–694.
Pradubsri, W., Chomtee, B., Borkowski, J.J., 2019. Using a genetic algorithm to generate d-optimal designs for mixture-process variable experiments. Qual. Reliab. Eng. Int. 35 (8), 2657–2676.
Rajaonarivony, M., Vauthier, C., Couarraze, G., Puisieux, F., Couvreur, P., 1993. Development of a new drug carrier made from alginate. J. Pharm. Sci. 82 (9), 912–917.
Rios, N., Lin, D.K.J., 2021. Order-of-addition mixture experiments. J. Qual. Technol., 1–10.
Sljivic-Ivanovic, M.Z., Smiciklas, I.D., Dimovic, S.D., Jovic, M.D., Dojčinović, B.P., 2015. Study of simultaneous radionuclide sorption by mixture design methodology. Ind. Eng. Chem. Res. 54 (44), 11212–11221.
Snee, R.D., Marquardt, D.W., 1974. Extreme vertices designs for linear mixture models. Technometrics 16 (3), 399–408.
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. American Statistical Association, Alexandria, VA, pp. 155–160.
Voelkel, J.G., 2019. The design of order-of-addition experiments. J. Qual. Technol. 51 (3), 230–241.
Voelkel, J.G., Gallagher, K.P., 2019. The design and analysis of order-of-addition experiments: an introduction and case study. Qual. Eng. 31 (4), 627–638.
Winker, P., 2000. Optimization Heuristics in Econometrics: Application of Threshold Accepting. Wiley.
Winker, P., Chen, J., Lin, D.K.J., 2020. The construction of optimal design for order-of-addition experiment via threshold accepting. In: Contemporary Experimental Design, Multivariate Analysis and Data Mining. Springer, pp. 93–109, Chapter 6.
Winker, P., Fang, K.-T., 1997. Application of threshold-accepting to the evaluation of the discrepancy of a set of points. SIAM J. Numer. Anal. 34 (5), 2028–2042.
Winker, P., Lin, D.K.J., 2011. Robust uniform design with errors in the design variables. Stat. Sin., 1379–1396.
Zhao, Y.L., Lin, D.K.J., Liu, M., 2021. Designs for order of addition experiments. J. Appl. Stat. 48 (8), 1475–1495.