Open access • Journal Article • DOI:10.1080/00401706.2015.1108233

Sliced Full Factorial-Based Latin Hypercube Designs as a Framework for a Batch Sequential Design Algorithm — Source link

Weitao Duan, Bruce E. Ankenman, Susan M. Sanchez, Paul J. Sanchez

Institutions: Northwestern University, Naval Postgraduate School

Published on: 31 Jan 2017 - Technometrics (Taylor & Francis)

Topics: Latin hypercube sampling, Orthogonal array, Sequential analysis, Design of experiments and Computer experiment

Sliced Full Factorial-Based Latin Hypercube Designs as a Framework for a Batch Sequential Design Algorithm

Weitao Duan[a], Bruce E. Ankenman[a], Paul J. Sanchez[b], Susan M. Sanchez[b]

[a]*Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL 60208*

[b]*Department of Operations Research, Naval Postgraduate School, Monterey, CA 93943*

Abstract

When fitting complex models, such as finite element or discrete event simulations, the experiment design should exhibit good properties of both projectivity and orthogonality. To reduce experimental effort, sequential design strategies allow experimenters to collect data only until some measure of prediction precision is reached. In this article, we present a batch sequential experiment design method that uses sliced Full Factorial-Based Latin Hypercube Designs (sFFLHDs), which are an extension to the concept of sliced Orthogonal Array-Based Latin Hypercube Designs (OALHD). At all stages of the sequential design, good univariate stratification is achieved. The structure of the FFLHDs also tends to produce uniformity in higher dimensions, especially at certain stages of the design. We show that our batch sequential design approach has good sampling and fitting qualities through both empirical studies and theoretical arguments.

Keywords:

Computer model, Latin hypercube design, sequential design, space filling design, metamodels, computer experiments, simulation experiments

---

1. Introduction

Computer simulations are frequently adopted in studying complex systems. For example, engineers use fluid dynamics models to visualize air flow around an aircraft and Monte Carlo simulations to optimize call center staffing. Although the power and speed of computers has grown significantly in the recent decade, a single evaluation of some computer models can take hours or even days. If the computer models are computationally expensive, metamodels, sometimes referred to as surrogate models, are constructed to approximate the complex computer models with sufficient accuracy. These metamodels replace the original computer models in optimization or "what if" analyses.

Building metamodels for these computer simulations involves sampling a set of points from the design space and fitting a model to the observed data. The focus of this paper will be on design of experiments which is used to select which set of points to sample from the design space. We will presume that kriging (or Gaussian process modeling), which has become widely used for building metamodels of complex deterministic computer experiments, will be used for the fitted model. Kriging, developed in geostatistics (Matheron 1963; Journel 1978), assumes spatial correlation between points. Responses at unobserved points are predicted using correlations between the observed points to create a response surface model. Recently Ankenman et al. (2010) extended kriging to the case of stochastic simulation. Although our approach is developed with kriging in mind, it is also appropriate for many other fitting methods, especially when there is little known about the true underlying surface.

A variety of experiment designs have been presented in literature for supporting kriging models. When the goal of the metamodel is to fully map the region of interest, designs utilize certain space-filling criteria and seek to place points in the design space uniformly. McKay et al. (1979) introduced Latin hypercubes for computer experiments where each level of each variable is sampled exactly once. This idea has spawned many variants.

Tang (1993) and Owen (1994) proposed the concept of orthogonal array-based LHD

(OALHD). An OALHD starts with an n-point OA of strength t for m columns ($t < m$), each at L levels, denoted $OA(n, m, L, t)$. For every t columns, the L^t level combinations appear the same number of times. OALHDs built on $OA(L^2, m, L, 2)$s have economical sample sizes and are used most widely. To construct an OALHD, the set of values from 1 to L^2 is partitioned into L groups: $\{1, \ldots, L\}, \{L+1, \ldots, 2L\}, \ldots, \{L(L-1), \ldots, L^2\}$. The levels in the OA correspond to each group. The levels in the OA in a given dimension are replaced by distinct integer values from its corresponding group. The replacement each time is random without replacement. OALHDs have good projectivity in any univariate and bivariate subspace if strength 2 OAs are used in construction.

Other space-filling criteria have also been adopted when constructing designs. Johnson et al. (1990) first defined the concept of minimax and maximin distance in the design of an experiment. The maximin criterion tries to maximize the minimum distance between any two points in the design. The minimax criterion minimizes the maximum distance between any nondesign point in the design space $\mathcal{S}$ and the closest design point in the design. Morris and Mitchell (1995) presented maximin LHDs which try to maximize the minimum distance between design points while maintaining the desirable projective properties of an LHD. Qian and Wu (2009) presented the idea of a sliced space-filling design. Each slice has good space-filling properties while the whole design achieves good uniformity in higher dimensional margins.

Sequential designs have gained popularity in recent research as experimenters desire the ability to terminate early if some stopping criterion is reached. The stopping criterion is usually based on an estimate of prediction variance or parameter estimation variance. In particular, in the search for a global optimizer, Bernardo et al. (1992) used an initial design to predict the response. If the predictor is not accurate, a subregion is chosen and explored. Otherwise, the objective is optimized using the current fitted model. Ranjan et al. (2008) presented sequential designs with the objective of contour estimation. Lam and Notz (2008)

proposed sampling additional points which maximize the expected improvement in model fit. Distance-based criteria also apply to the construction of sequential designs. Besides maximin and minimax criteria, Johnson et al. (1990) examined a weighted distance criterion for choosing new design points.

Recently, Loeppky et al. (2010) introduced the notion of batch sequential designs for computer experiments, in particular the bin-based sequential design. The sequential bin structure is established by a set of defining relations. The bins are used to construct augmenting sets of runs that yield, as nearly as possible, aggregate designs that are Latin hypercube sampling (LHS) with near maximin distance at each batch stage. A batch sequential experimental design allows the experimenter to successively add batches of design points to an experiment. The goal is that after any batch is added, the design has reasonably good projectivity and orthogonality properties. The stopping criterion can be invoked when the desired precision is reached.

In this paper, we present a batch sequential experiment design that uses the idea of sliced space-filling designs from Qian and Wu (2009) and extends the work of Loeppky et al. (2010). Like Loeppky et al. (2010)'s bin-based designs, our design possesses good orthogonality and projectivity at intermediate stages and leads to an OALHD. However, our design does not require preselection of a total number of runs. Instead, it allows for batches to be added indefinitely. At certain stages of the design, which we call the *golden stages*, our design achieves very special space-filling properties.

We now introduce the definition of a full factorial-based Latin hypercube design (FFLHD), which our sequential design achieves at the golden stages. A D-dimensional n-point design $\mathbf{X}$ with L levels is said to be an L-level FFLHD if two properties hold. First, when every dimension of $\mathbf{X}$ is partitioned into L evenly spaced levels of $(0,1]$:$(0,1/L], (1/L, 2/L], \ldots, ((L-1)/L, 1]$, the resulting design is an L-level full factorial design. Second, when $\mathbf{X}$ is projected onto any dimension, precisely one point falls within n equally spaced levels given by

$(0, 1/n], (1/n, 2/n], \ldots, ((n-1)/n, 1]$.

At each batch stage, one slice from an FFLHD is sampled. Therefore, we call our sequential design a sliced full factorial-based Latin hypercube design (sFFLHD). Three design matrices are created in the process of sequential sampling: the big grid design, the intermediate grid design, and the small grid design. The big grid design preserves orthogonality, while the small grid design preserves LHD projectivity. As the sequential design adds a large number of design points, orthogonality on L levels becomes a weak criterion. The intermediate grid design allows the sequential design to build orthogonality on more than L levels. At the golden stages, the sequential design achieves its best space-filling properties and is an FFLHD.

The remainder of this paper is organized as follows. In Section 2 we introduce some notation, provide an example of an sFFLHD, and present a general method for constructing an sFFLHD. In Section 3 we derive some theoretical properties of sFFLHDs. In Section 4 we compare the results obtained using different design procedures for several numeric examples and propose some choices of stopping criteria for sFFLHD. In Section 5 we demonstrate an application of sFFLHD to a logistics simulation model. We summarize our work and present our conclusions in Section 6.

2. Sliced Full Factorial-based Latin Hypercube Design Construction

We begin with some notation.

General matrix notation

$\mathbf{S}_{\cdot j}$: jth column of a matrix $\mathbf{S}$
$\mathbf{S}_{i\cdot}$: ith row of a matrix $\mathbf{S}$
$\mathbf{S}_{[i:j,\cdot]}$: Rows i to j of a matrix $\mathbf{S}$
$\mathbf{S}_{ij}$: Element in the ith row and jth column of a matrix $\mathbf{S}$

Parameters that are constant throughout sFFLHD

L: Levels of the big grid and also the batch size
D: Dimension (number of factors)

Parameters that update after each batch of sFFLHD

b: Batch number
l_b: Levels of the small grid after b batches
L_b: Levels of the intermediate grid after b batches
n_b: Number of sampling points after b batches, $n_b = bL$

Matrices used in sFFLHD

$\mathbf{A}^r$: rth $OA(L^2, D, L, 2)$ such that $\mathbf{A}^r$'s are non-overlapping
$\mathbf{A}^r_p$: pth slice of $\mathbf{A}^r$
a^r_{pij}: jth element of ith row in $\mathbf{A}^r_p$
$\mathbf{X}_b$: Sequential design after b batches, x^r_{pij}'s are elements of $\mathbf{X}_b$
$\mathbf{V}_b$: Small grid design matrix after b batches, v^r_{pij}'s are elements of $\mathbf{V}_b$
$\mathbf{M}_b$: Intermediate grid design matrix after b batches, m^r_{pij}'s are elements of $\mathbf{M}_b$
$\mathbf{W}_b$: Big grid design matrix after b batches, w^r_{pij}'s are elements of $\mathbf{W}_b$
$\mathbf{V}_{.d}$: Set of levels in the small grid that have been observed in dimension d after b batches ($d = 1, 2, \ldots, D$), also the dth column of $\mathbf{V}_b$

To construct an sFFLHD, we consider a special type of orthogonal array $OA(n, m, L, t)$, $t = 2$, $n = L^2$. For any two columns, all level combinations appear exactly once. Equation (1) shows an $OA(9, 4, 3, 2)$, which is an example of this type of orthogonal array.

$$\mathbf{Z} = \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 2 \\ 0 & 2 & 2 & 1 \\ \hdashline 1 & 0 & 2 & 2 \\ 1 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \\ \hdashline 2 & 0 & 1 & 1 \\ 2 & 1 & 2 & 0 \\ 2 & 2 & 0 & 2 \end{array}\right] \tag{1}$$

If we partition $\mathbf{Z}$ to three slices by the first column, that is, rows 1-3, 4-6, 7-9 are the three slices, columns 2-4 of each slice form a different Latin hypercube. In fact, an $OA(L^2, D+1, L, 2)$ can always be partitioned into L slices of D-dimensional Latin hypercubes by using one column to separate the slices.

The batches (slices) of the sFFLHD design are constructed using a series of orthogonal arrays $A^1, A^2, \ldots$ with strength $t (t \geq 2)$. Each orthogonal array is a non-overlapping fraction of a full factorial design of L levels and D factors. The union of $A^1, A^2, \ldots$ forms the big grid design. A^i is partitioned into L slices as demonstrated above, and provides L batches for the big grid design. For more detail, consider the following example:

Run	Batch	$\mathbf{X}_9 =$			$\mathbf{V}_9 =$			$\mathbf{W}_9 = \mathbf{M}_9 =$				
1		0.281	0.172	0.193	8	5	6	0	0	0		
2	1	0.343	0.400	0.695	10	11	19	1	1	2		$\mathbf{A}_1^1$
3		0.681	0.953	0.403	19	26	11	2	2	1		
4		0.149	0.885	0.878	5	24	24	0	2	2		
5	2	0.648	0.303	0.575	18	9	16	1	0	1	$\mathbf{A}^1$	$\mathbf{A}_2^1$
6		0.922	0.662	0.034	25	18	1	2	1	0		
7		0.062	0.472	0.483	2	13	14	0	1	1		
8	3	0.484	0.747	0.315	14	21	9	1	2	0		$\mathbf{A}_3^1$
9		0.810	0.036	0.987	22	1	27	2	0	2		
10		0.128	0.213	0.748	4	6	21	0	0	2		
11	4	0.587	0.413	0.547	16	12	15	1	1	1		$\mathbf{A}_1^2$
12		0.957	0.991	0.262	26	27	8	2	2	0		
13		0.077	0.676	0.442	3	19	12	0	2	1		
14	5	0.545	0.134	0.109	15	4	3	1	0	0	$\mathbf{A}^2$	$\mathbf{A}_2^2$
15		0.972	0.535	0.832	27	15	23	2	1	2		
16		0.322	0.607	0.049	9	17	2	0	1	0		
17	6	0.462	0.805	0.713	13	22	20	1	2	2		$\mathbf{A}_3^2$
18		0.772	0.050	0.650	21	2	18	2	0	1		
19		0.001	0.266	0.356	1	8	10	0	0	1		
20	7	0.629	0.486	0.250	17	14	7	1	1	0		$\mathbf{A}_1^3$
21		0.712	0.849	0.785	20	23	22	2	2	2		
22		0.217	0.913	0.145	6	25	4	0	2	0		
23	8	0.376	0.092	0.917	11	3	25	1	0	2	$\mathbf{A}^3$	$\mathbf{A}_2^3$
24		0.866	0.578	0.625	24	16	17	2	1	1		
25		0.226	0.341	0.948	7	10	26	0	1	2		
26	9	0.431	0.726	0.478	12	20	13	1	2	1		$\mathbf{A}_3^3$
27		0.848	0.250	0.180	23	7	5	2	0	0		

Figure 1: First 9 batches of an sFFLHD when $D = L = 3$, with the associated small grid, intermediate grid, and large grid design matrices.

Example 1. $D = L = 3$. Figure 1 shows the first 9 batches (27 runs) of an sFFLHD design $\mathbf{X_9}$, along with the associated small grid design $\mathbf{V_9}$ and large grid design $\mathbf{W_9}$. In the first 27 runs, $\mathbf{M}$ has $L_b = 3$ levels/dimension and $\mathbf{V}$ has $l_b = 27$ levels/dimension. In the next 27 runs, $\mathbf{M}$ has $L_b = 9$ levels/dimension and $\mathbf{V}$ has $l_b = 81$ levels/dimension. Figure 2 provides the intermediate design $\mathbf{M}_{18}$ and the small grid design $\mathbf{V}_{18}$. $\mathbf{X}$ is the experiment design where the factors have levels scaled from 0 to 1.

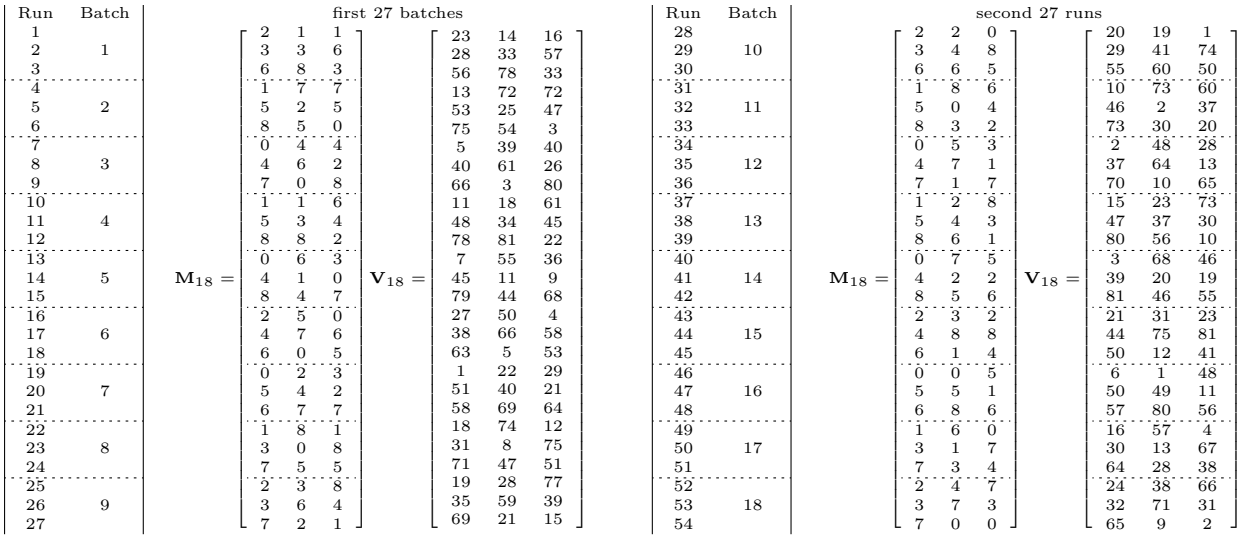

Figure 2: sFFLHD for $D = L = 3$: intermediate grid design after 18 batches, small grid for second 9 batches

Our algorithm converts batches of the big grid design into batches of the intermediate and small grid designs, while preserving both orthogonality on the big grid scale and LHD-like projectivity properties on the small grid scale. The intermediate grid design ensures that the sequential design is a subset of a full factorial design on the intermediate grid scale. After each batch is added, the small grid design always remains a subset of an OALHD.

When enough batches have been added so that the big grid design is a full factorial design with L levels, the small grid is a Latin hypercube with n_b levels.

At a high level, this algorithm observes batches of L design points sequentially until a stopping criterion is reached or an L-level L^D-point FFLHD is constructed, which we call a golden stage. If experimentation is to continue beyond the golden stage, then a new intermediate grid design with $(aL)^D$ design points is created ($a^q = L, q \in \mathbb{N}_+$). Batches of L design points are then observed sequentially until an aL-level $(aL)^D$-point FFLHD is created, which is the next golden stage. This process can continue indefinitely so the design at any point in the process is a subset of an $a^{sD}L^D$-point ($s \in \mathbb{N}$) a^sL-level FFLHD. Each batch is guided by an orthogonal array at the big grid level and is a non-overlapping subset of the $a^{sD}L^D$-point LHD. This preserves some measure of orthogonality and projectivity after

each batch.

The sFFLHD is generated with the following algorithm: (comments are in italic)

Step 0: Initialize $b = 0$, $l_0 = L$, $L_0 = L$, $n_0 = 0$.

Step 1: Choose batch size L such that $OA(L^2, D+1, L, 2)$ exists. Randomly permute rows and columns of the orthogonal array and then sort it by the first column and denote the other D columns by $\mathbf{A}^1$. Slices of $\mathbf{A}^1$ are determined by the sorted column. In this fashion, $\mathbf{A}^1$ is an $OA(L^2, D, L, 2)$ and each slice is a Latin hypercube design with L levels and will be used as a batch of the sequential design. Create $L^{D-2} - 1$ non-overlapping OAs called $\mathbf{A}^2, \ldots, \mathbf{A}^{L^{D-2}}$, which can be generated from $\mathbf{A}^1$ (see Appendix A).

Step 2:
For each $\mathbf{A}^r$, $r = 1, 2, \ldots, L^{D-2}$
For each p, $p \in \{1, \ldots, L\}$, $\mathbf{A}^r_p$ is the pth slice of $\mathbf{A}^r$,
Update the level of the small grid design if necessary:
If $n_b + L > l_b$, $\mathbf{X}_b$ reached an LHD,
then update small grid levels by $l_b = al_b$, where $a^q = L, q \in \mathbb{N}_+$. The small grid design $\mathbf{V}_b$ is updated by $\mathbf{V}_b = \lceil \mathbf{X}_b * l_b \rceil$.
Otherwise, continue.

Let $\mathbf{G}_{[n_b+1:n_b+L,.]} = \mathbf{A}^r_p$, and $\epsilon_{[n_b+1:n_b+L,.]}$ be an L by D matrix of random uniform number from $[0, 1)$.
Add batch $b+1$ using the function defined in Appendix B:
$NB(\mathbf{G}_{[n_b+1:n_b+L,.]}, \epsilon_{[n_b+1:n_b+L,.]}, b)$

Observe batch $b+1$
If the stopping criterion is satisfied
then EXIT
else continue
$b = b + 1$
$n_b = n_{b-1} + L$
Next p
Next r

Step 3: *If **Step 2** completes before the stopping criterion is met, then the intermediate grid design is a full factorial design with L_b levels.*

Update the number of levels of the intermediate grid design to $L_b = aL_b$ and update $\mathbf{M}_b$ to $\mathbf{M}_b = \lfloor \mathbf{X}_b * L_b \rfloor$. *Now $\mathbf{M}_b$ is a^{-D} fraction of a full factorial design with L_b levels.*

Step 4: Let $\mathbf{v}$ be a D length vector with integer values from 0 to $a-1$ and $\mathbf{v} \neq \mathbf{0}$. There are $a^D - 1$ possible unique $\mathbf{v}$'s. Let $f, f \in 1, \ldots, a^D - 1$, be the element index after randomly permute the elements of $\{\mathbf{v}\}$. Pick a non-overlapping fraction of the full factorial design with L_b levels $\mathbf{F}_1$ by $F_{1,ij} = a\lfloor M_{ij}/a \rfloor + [(M_{ij} + v_1(j)) \bmod a]$ for all i, j.

Different $\mathbf{v}_f$*'s result in different* $\mathbf{F}_f$*'s, so the* $\mathbf{F}_f$*'s are non-overlapping fractions of the* L_b *level full factorial. Note that when we project* $\mathbf{F}_f$ *into* L *levels, we get a full factorial or replicates of full factorial in* L *levels* $\{\mathbf{H}_k, i = 1, \ldots, (L_b/aL)^D\}$.

Slice each $\mathbf{H}_k$ into L^{D-2} non-overlapping OAs.

The set of all OAs after slicing is $\mathbf{F}^r$, $r = 1, \ldots, L^{D-2}(L_b/aL)^D$. $\mathbf{F}^r$ *is an* $OA(L^2, D, L, 2)$ *when projected into* L *levels.* $\mathbf{F}^r_p$ *is the* p*th slice of* $\mathbf{F}^r$*, similar to the relationship between* $\mathbf{A}^r_p$ *and* $\mathbf{A}^r$.

For each p,

Update the level of the small grid design if necessary:
If $n_b + L > l_b$,
then $l_b = al_b$, and $\mathbf{V}_b = \lceil \mathbf{X}_b * l_b \rceil$.
Otherwise, continue.

Let $\mathbf{G}_{[n_b+1:n_b+L,.]} = \mathbf{F}^r_p$, and let $\epsilon_{[n_b+1:n_b+L,.]}$ be an L by D matrix of random uniform number from $[0, 1)$.
Perform $NB(\mathbf{G}_{[n_b+1:n_b+L,.]}, \epsilon_{[n_b+1:n_b+L,.]}, b)$ and update all design matrices accordingly.

Observe batch $b + 1$
If the stopping criterion is satisfied
then EXIT
else continue
$b = b + 1$
$n_b = n_{b-1} + L$
Next p

Step 5: If $\mathbf{M}_b$ is a full factorial in L_b levels, update L_b to $L_b = aL_b$ and $\mathbf{M}_b$ to $\mathbf{M}_b = \lfloor \mathbf{X}_b * L_b \rfloor$. Repeat **Step 4** until the stopping criterion is met. □

Figures 1 and 2 show how batches and runs are constructed for an example where $D = L = 3$. In the rightmost columns of Figure 1, we show the non-overlapping OAs $\mathbf{A}^i$'s that are used in the big grid design, $\mathbf{W}$ for the first 9 batches. The small grid design $\mathbf{V}$, the intermediate grid design $\mathbf{M}$, and the sequential design $\mathbf{X}$, are constructed using the above algorithm. In the first 27 runs (see Figure 1), $\mathbf{M}$ has $L_b = 3$ levels/dimension and $\mathbf{V}$ has $l_b = 27$ levels/dimension. In Figure 2, we add another 9 batches. Now, $\mathbf{M}$ has $L_b = 9$ levels/dimension and $\mathbf{V}$ has $l_b = 81$ levels/dimension. The intermediate grid design of runs 28-54 is constructed using $\mathbf{v}_1 = [0, 1, 2]$. $\mathbf{X}$ (not shown) is the experiment design where the factors have levels scaled from 0 to 1.

3. Analysis of sFFLHD for mean estimation

The mean estimator of a design provides information on the average response of the design space. A good estimator should achieve good accuracy and precision. In a kriging setting, little is usually known about the form of the response surface. Thus, estimating the surface can be thought of as estimating the mean over small regions. The ability to estimate the response surface is related to the ability to estimate a mean of a given subregion. Since our design methodology essentially continues to fill the space in a uniform way, in the limit, any subregion will begin to be filled as if it were the only region of interest. In this section, we will show the mean estimator from an sFFLHD achieves good variance reduction compared to random sampling, especially at certain stages.

3.1. Derivation of Mean Estimator of sFFLHD

Let dF denote the uniform probability measure on $(0,1]^D$. The true average output of a measurable function f in $(0,1]^D$ with $\int f(x)^2 dF < \infty$ can be expressed as $\mu = \int_{(0,1]^D} f(\mathbf{x}) dF$. Consider an experiment with n runs labeled as $\{\mathbf{x_i}\}, i = 1, 2, \ldots, n$, where $\mathbf{x_i} = (x_1, x_2, \ldots, x_D)$. The sample mean $\overline{Y}$ of n experiment runs is used as a predictor for μ by $\overline{Y} = \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x_i})$ μ is then estimated by $\overline{Y}$. The quality of $\overline{Y}$ depends on its mean and variance.

Let $\mathcal{D}$ denote the power set of $C = \{1, 2, \ldots, D\}$ and $dF_u = \prod_{i \in u} dx_i$ denote a uniform measure on $(0,1]^{|u|}$,$u \in \mathcal{D}$. The ANOVA decomposition of f (Owen 1994) is given by $f = \sum_{u \in \mathcal{D}} \alpha_u$.

The components α_u are defined inductively via

$$\alpha_u = \int (f - \sum_{v \subset u} f_v) dF_{C \setminus v}.$$

$\alpha_\varnothing$ represents the grand mean. $\alpha_i = \int (f - \alpha_\varnothing) dF_{C \setminus i}$ is the main effect of dimension i, and so on. With $\int \alpha_u \alpha_v dF = 0, u \neq v$, $\int f^2 dF$ can be decomposed into $\sum_{u \in \mathcal{D}} \int \alpha_u^2 dF$, while the variance $\sigma^2 = \int (f - \mu) dF$ is simply $\sum_{u \in \mathcal{D} \setminus \varnothing} \int \alpha_u^2 dF$.

Let Z_{l_b} denote the set $\{1, 2, \ldots, l_b\}$. Proposition 1 establishes the forms of the marginal and joint probability mass functions.

Proposition 1. *Let $\mathbf{V}^B$ denote the small grid design of batch B. For every element in $\mathbf{V}^B$, v_{ij}^B,*

$$P(v_{ij}^B) = \frac{1}{l_B},\ s \in Z_{l_B}.$$

for every i, k, where $i \neq k$

$$P(v_{ij}^r = s, v_{kj}^r = t) = \begin{cases} \frac{L}{l_B^2(L-1)} & s, t \in Z_{l_B} \text{ and } \lfloor sL/l_B \rfloor \neq \lfloor tL/l_B \rfloor \\ 0 & \text{otherwise} \end{cases}$$

for $B_1 \neq B_2$, and $b = max\{B_1, B_2\}$ we have

$$P(v_{ij}^{B_1} = s, v_{kj}^{B_2} = t) = \begin{cases} \frac{1}{l_b^2} & s, t \in Z_{l_b} \text{ and } \lfloor sL/l_b \rfloor \neq \lfloor tL/l_b \rfloor \\ \frac{1}{l_b(l_b - L)} & s, t \in Z_{l_b},\ s \neq t \text{ and } \lfloor sL/l_b \rfloor = \lfloor tL/l_b \rfloor \\ 0 & \text{otherwise.} \end{cases}$$

□

Following the definition in Owen (1994), let W_{ij} denote the jth entry of ith row of the big grid design $\mathbf{W}$. For $u \subset \mathcal{D}$, let $\omega_{ij}(u) = \{k \in u : W_{ik} = W_{jk}\}$ and $\omega_{ij}^B(u) = \{k \in u : W_{ik}^B = W_{jk}^B\}$, and define

$$M(u, r) = \sum_{i=1}^{n} \sum_{j=1}^{n} 1\{|\omega_{ij}(u)| = r\} \quad \text{and} \quad M^B(u, r) = \sum_{i=1}^{L} \sum_{j=1}^{L} 1\{|\omega_{ij}^B(u)| = r\}.$$

Owen (1994) showed that variance of the mean estimator from an n-point lattice sampling design can be written in the following form:

$$Var(\overline{Y}) = n^{-2} \sum_{|u| \geq 2} \sum_{r=0}^{|u|} M(u, |u|)(1 - L)^{r - |u|} Var(\alpha_u(\mathbf{x})) + o(n^{-1})$$

Using the probability mass functions in **Proposition 1**, we can derive the expectation and variance of the mean estimator of sFFLHD.

Proposition 2. *Let $\overline{Y^B} = \sum_{i=1}^{L} f(\mathbf{X}_{i.}^B)$, the mean estimator using a single batch of sFFLHD. Then*

$$E(\overline{Y^B}) = \mu \quad \text{and} \quad E(\overline{Y}) = \mu. \tag{2}$$

This shows that the mean estimator of each batch and the whole sequential design at any batch stage is unbiased.

For smooth function f, as $L \to \infty$

$$Var(\overline{Y^B}) = \sum_{|u|\geq 2} M^B(u,|u|)L^{-2}Var(\alpha_u(\mathbf{x})) + o(L^{-1}). \tag{3}$$

At stages where the big grid design is an OA, as $L \to \infty$ we also have

$$Var(\overline{Y}) = \sum_{|u|\geq 3}\sum_{r=0}^{|u|} M(u,|u|)l_b^{-2}Var(\alpha_u(\mathbf{x})) + o(l^{-1}). \tag{4}$$

At stages where $L_b^D = n$ the sequential design is an FFLHD, as $n \to \infty$ we have

$$Var(\overline{Y}) = O(L_b^{-D-2}). \tag{5}$$

□

From **Proposition 2**, we can see that the variance reduction achieved by each batch of our procedure, compared to random sampling, is similar to that achieved by an ordinary Latin hypercube design. Greater variance reduction is achieved when the sequential design is an OALHD as pointed out in He and Qian (2011). At other stages, the sequential design can be thought of as an LHD with uneven levels, where points do not spread out evenly when projected onto a dimension. This is different from an ordinary LHD where, in each dimension, levels correspond to equal-width intervals. However, with the structure of big grid designs and grid designs, we attain good sampling properties even at these intermediate stages. We demonstrate this in the next section through empirical studies.

4. Numerical Examples

4.1. Comparison of Gaussian Model Fitting against MmDist

In this section, we focus on the comparison of our sFFLHD with a maximin distance sequential design (MmDist), as this seems to be the most widely used sequential space-filling

design. Comparisons with other designs are summarized later. Suppose the batch size is L. An MmDist design starts with a maximin LHD with L points, and each subsequent point is placed to maximize the minimum interpoint Euclidean distance. Although MmDist is a fully sequential design, we can group sets of L points into batches and implement the design in a batch sequential manner.

We first compare sFFLHD with the fully sequential maximin design for estimating two well-known test functions, the borehole and Rastrigin function. We then compare the designs for estimating surfaces generated from a Gaussian random process model.

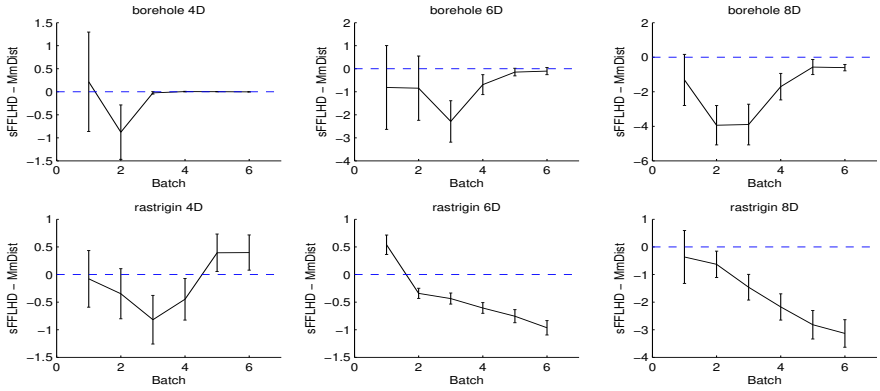

Figure 3: Borehole and Rastrigin Examples: RMSE differences between sFFLHD and MmDist (Batches 1–6)

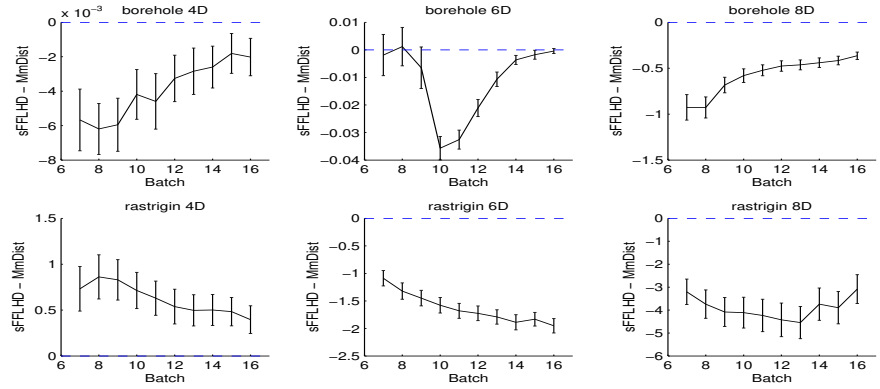

Figure 4: Borehole and Rastrigin Examples: RMSE differences between sFFLHD and MmDist (Batches 7–16)

Example 2. Borehole Example

(Worley 1987) used a model to demonstrate the flow of water through a borehole. The

model has eight input variables. All designs have been scaled to fit the range of interest. In our comparison, we vary 4, 6 and all of the eight variables. We sample 8 points at a time. The final budget is set at 128 runs. For each design method, function values are evaluated at design points and a GP model is fit at each batch stage. A 10,000-point maximin LHD is used to assess the RMSE of each GP model. Confidence intervals of RMSE differences are obtained via 100 independent replications. Since the difference is (RMSE for sFFLHD)−(RMSE for MmDist), a confidence interval that is completely negative indicates better performance of the sFFLHD. In order to facilitate proper scaling for visualization, Batches 1–6 are shown in Figure 3 and Batches 7–16 in Figure 4. From the first row of plots in Figures 3 and 4, we can see that MmDist is as good as sFFLHD in terms of RMSEs in the 4-dimensional case since almost all the confidence intervals contain zero. In the 6 and 8-dimensional cases, the confidence intervals are almost always negative, so we conclude that sFFLHD produces significantly lower RMSEs across all batch stages.

Example 3. Rastrigin Function

$$f(x) = 10D + \sum_{i=1}^{D} [x_i^2 - 10cos(2\pi x_i)]$$

D is the dimensionality of the Rastrigin function and $x_i \in [-5.12, 5.12]$ in each dimension. We use $D = 4, 6$ and 8 to compare the two designs and scale the Rastrigin function to fit $[0, 1]^D$. A batch of 8 design points are sampled each time and the final batch is set at 128 points. For each design method, 100 independent designs are generated. A 10,000-point maximin Latin hypercube design is used to compare the RMSEs at each batch stage. Confidence intervals of RMSE differences are obtained. The second row of plots in Figure 3 and 4 shows the RMSE differences between two designs with batch 1-6 in Figure 3 and 7-16 in Figure 4. sFFLHD performs much better than MmDist on RMSEs in 6 and 8-dimensional cases. For 4-dimensional case, differences between the two designs are not significant at early

stages; however, MmDist dominates sFFLHD in the late stages.

Example 4. Gaussian Process Model

For this test problem, we consider several k-dimensional Gaussian processes (k=2,4,6 and 8). Different θ values represent different scenarios. Data are generated on 10000-point maximin distance design in $[0,1]^k$. The model fitted from the 10000 points can well capture the true response surface. We sample 8 points at a time. The final budget is set at 128 runs. For each design method, function values are evaluated at design points and a GP model is fit at each batch stage. The 10000 points are used to calculate the RMSE of each GP model. Confidence intervals of RMSE differences are obtained via 100 independent replications.

First, θ is set to be 5 in each dimension. The true Gaussian surface is relatively smooth. The second rows of Figures 5 and 6 show that in high dimensional cases, RMSEs of sFFLHD are favorable compared to maximin distance design at almost all stages. In low dimensional cases, maximin distance design is comparable to or better than sFFLHD, because maximin distance design tends to have points spread out evenly in low dimensions.

We increase θ to 15 in each dimension. The simulated Gaussian surfaces are now relatively non-smooth. For non-smooth surfaces, sFFLHD performs better than MmDist in 4 dimensional and higher cases (see first rows of Figures 5 and 6). However, the size of the advantage over MmDist diminishes when fitting rough surfaces because neither design is able to capture the true response well with a small number of design points.

4.2. Comparison with Other Designs

We also compare sFFLHD with other design methods in addition to MmDist design. Maximin LHD (MmLHD) is a widely used space-filling design. To implement it in a batch sequential manner, a MmLHD of the same final budget is generated in each replication and then divided into batches of the same size as in the examples. We call this design batch sequential MmLHD (bMmLHD). Even though bMmLHD cannot go beyond the final

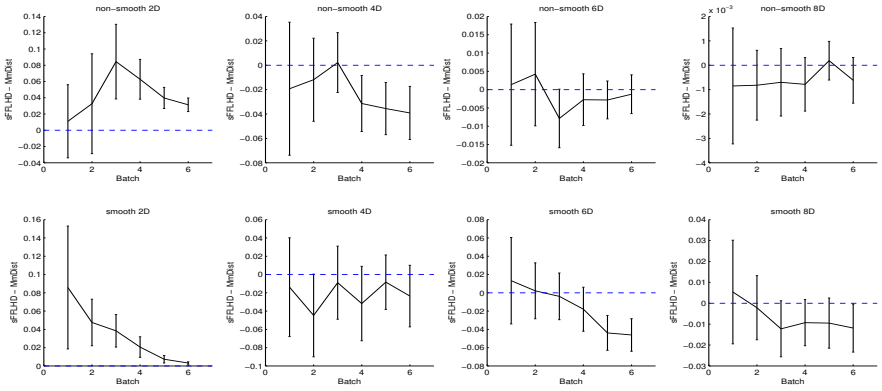

Figure 5: Gaussian Example: RMSE difference between sFFLHD and MmDist (Batch 1–6)

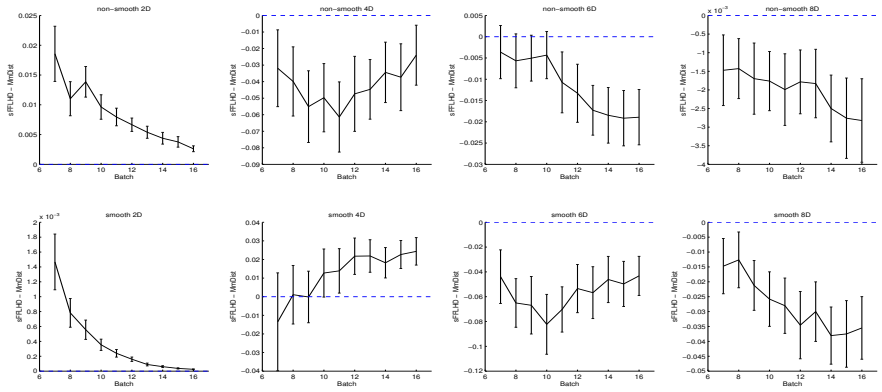

Figure 6: Gaussian Example: RMSE difference between sFFLHD and MmDist (Batch 7–16)

budget (which is often times unknown a priori), it serves as a baseline for RMSE comparison. Another design method, batch sequential LHD (bLHD) simply uses random LHDs of the same size as batches. This may be appealing due to its light computational requirements, but this design does not spread points evenly on a finer scale when projected onto any single dimension. A random version of sFFLHD (rsFFLHD) is also included in the comparison to demonstrate the value of having each batch be an LHD. To create the rsFFLHD, we use a slightly modified version of the sFFLHD algorithm. In Step 1 of sFFLHD, rows of $\mathbf{A}^1$, $\mathbf{A}^2$,..., are shuffled, so that the big grid design of each batch of rsFFLHD may not be a LHD. However, at the end of Step 2, the big grid design of rsFFLHD remains an OALHD.

We use Examples 2, 3, 4 to compare the above design methods with sFFLHD. The average RMSE differences are listed in Appendix D. sFFLHD performs better than bMmLHD during

early-stages and mid-stages, and as well at final stage in terms of RMSE. In comparison with bLHD, performance of bLHD is never better than sFFLHD. Close to the stages where the big grid design of rsFFLHD is an OALHD, rsFFLHD and sFFLHD performs equally well as expected. However, sFFLHD performs better than rsFFLHD at other stages leading us to conclude that forcing each batch to be an LHD produces better space-filling property at stages when sFFLHD is not an OALHD.

4.3. Stopping Criteria

The most important attribute of sFFLHD is the ability to stop at any batch stage while maintaining good space-filling properties. While smaller RMSE of fit is often desirable, computing actual RMSE requires the knowledge of the true model which is not known in most cases. However, the emulator often enables us to estimate the MSE of prediction at some unobserved point. For instance, if GP model is used as the emulator, given the GP model, the predicted MSE for an unobserved site $\mathbf{x}$ can be computed from the following expression:

$$MSE(Y(\mathbf{x})) = \sigma^2(1 - \mathbf{r}^T(\mathbf{x})\mathbf{R}^{-1}\mathbf{r}(\mathbf{x})) \tag{6}$$

where $\mathbf{r}(\mathbf{x}) = [\mathbf{R}(\mathbf{x}, \mathbf{x}_1), \ldots, \mathbf{R}(\mathbf{x}, \mathbf{x}_n)]$ and the correlation function $\mathbf{R}(\mathbf{x}, \mathbf{x}') = \exp(-\sum_{i=1}^{d} \theta_i(x_i - x_i'))$. The root integrated MSE (RIMSE)

$$RIMSE = \sqrt{\int_{\mathcal{S}} MSE(Y(\mathbf{x})d\mathbf{x}} \tag{7}$$

can be used as a measurement of uncertainty for prediction. Typically, the parameters $\theta_i, \forall i$ are estimated and then RIMSE is approximated by computing the estimated MSE at each point on a large grid and taking the root of the average across the grid.

Cross validation also can provide a performance measure of the GP model. Leave-one-out cross validation is often preferred as only one observation is left out for each cross validation

and cross validation fits are close to the fit with all data. An example is studied in Section 5 to demonstrate the usage of the above stopping criteria.

4.4. Comparison of Mean Estimators

In this section, we compare the properties of the mean estimators of 4 different design methods(sFFLHD, bMmLHD, MmDist and rsFFLHD).

Example 5. Suppose the computer model is given by

$$f = x_1 + x_2 + x_1x_2 + x_1^2 + x_2^2 + min(e^{3x_2}, 10) - 1.5x_1x_2x_3 + x_3^2$$

$$x_1 \sim Unif[-2, 0], x_2 \sim Unif[0, 1], x_3 \sim Unif[0.5, 1.5]$$

We adopt a final run size of 64 using batches of size 4 and calculate $\hat{\mu}$ for each scheme at each batch stage over 2,000 replications. RMSE of $\hat{\mu}$ at selected batch stages are shown in Table 1. The result shows that in terms of RMSE, sFFLHD has the best mean estimator at all batch stages among the compared design methods. Especially at stages where sFFLHD is an OALHD where numbers are in bold, the mean estimator of sFFLHD is significantly superior to all the designs except rsFFLHD, however, at these stages sFFLHD and rsFFLHD are equivalent designs.

Batch	1	**4**	8	12	**16**
Design Points	4	**16**	32	48	**64**
sFFLHD	0.214	**0.005**	0.015	0.004	**0.000069**
bMmLHD	5.001	**0.926**	0.275	0.059	**0.000076**
MmDist	0.235	**0.129**	0.072	0.033	**0.0379**
rsFFLHD	3.371	**0.006**	0.014	0.075	**0.000070**

Table 1: Function in Ex. 5: comparison of RMSE of $\hat{\mu}$ for each design scheme

Example 6. Borehole Example

All eight dimensions are used in this example. For each design method, a final run size of 128 with batches of size 8 are used. Function values are evaluated at design points and

the sample mean is calculated at each batch stage. RMSEs of $\hat{\mu}$ are obtained via 2,000 independent replications. Table 2 summarizes the result. Similar to the result in Example 5, the mean estimators of sFFLHD are superior to other designs, except when equivalent to rsFFLHD.

Batch	1	4	**8**	12	**16**
Design Points	8	32	**64**	96	**128**
sFFLHD	20.645	1.546	**0.059**	0.310	**0.027**
bMmLHD	254.137	49.674	**13.120**	5.995	**0.709**
MmDist	15.526	23.102	**10.886**	6.997	**5.361**
rFFLHD	223.847	34.909	**0.069**	3.502	**0.024**
bLHD	14.887	2.944	**1.660**	1.036	**0.888**

Table 2: Borehole function: comparison of RMSE of $\hat{\mu}$ for each design scheme

5. Application: Operational Availability Simulation

Our final example applies sFFLHD to a discrete-event simulation for logistics operations. The basic scenario is that an operational unit begins with a fleet of working vehicles. Over time, vehicles break down and are repaired, or become due for scheduled maintenance and are serviced. Of interest is the availability of vehicles at the beginning of each day, since this determines what operations can be conducted. The proportion of the initial fleet available is called the operational availability, and within the U.S. Department of Defense this is abbreviated as "Ao."

We will provide a description of the model logic using an event graph, which is a pictorial representation of discrete event simulation model from a state transition perspective. Each event (i.e., a vehicle breakdown, the start or end of scheduled maintenance or repair, etc.) is represented by a vertex where state transitions can occur. A quintessential event graph is shown in Figure 7. Here, **A** and **B** are events, t is a delay (which could be constant, random, or some function of the state), and **c** is a boolean function of the state. As Sanchez (2006) describes, this event graph can be readily translated into English as follows:

"When event **A** occurs, first perform all of its state transitions. Then, if boolean condition **c** is true, schedule event **B** to occur t time units later."

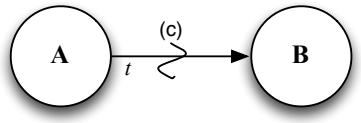

Figure 7: The Quintessential Event Graph

With these principles, details of the model logic are shown in the event graph of Figure 8. Only three state variables change over time:

- Q_{maint}: the number of vehicles in the queue awaiting scheduled maintenance service,
- Q_{repair}: the number of vehicles in the queue awaiting repair after a breakdown, and
- S: the number of maintenance personnel (servers) available.

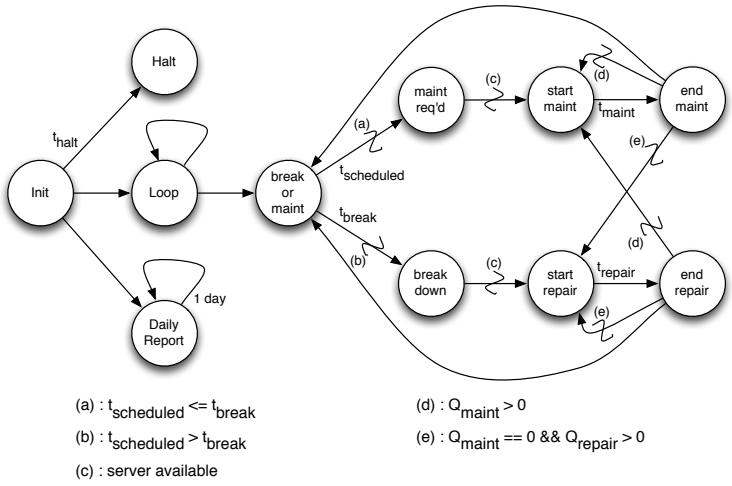

Figure 8: Event graph of Ao (operational Availability) simulation model

The "Initialize" node sets the initial state of the system; it also schedules the simulation halt at t_{halt} in the future, initiates the loop for printing daily reports of the number of vehicles available at the beginning of each day, and then loops over each vehicle in the fleet

to determine the next event, either a breakdown or a scheduled maintenance event. The time until the next breakdown (t_{break}) is exponentially distributed, while scheduled maintenance occurs at a fixed time t_{sched} in the future. If $t_{sched} \leq t_{break}$ then the vehicle will join the queue for maintenance and the length of this queue (Q_{maint}) is increased by one. Once a server is available, the vehicle starts service, Q_{maint} is decremented by 1, and the time at which the vehicle will be fully serviceable is scheduled t_{maint} in the future. Here, t_{maint} is a random variable drawn from a uniform distribution (with high probability) or a Weibull distribution (with low probability), representing the event that a previously undiagnosed breakdown is identified during scheduled maintenance. Once the maintenance finishes, the server becomes free. If $Q_{maint} > 0$ the server will immediately begin working on the next vehicle awaiting scheduled maintenance; if $Q_{maint} = 0$ but $Q_{repair} > 0$ the server will begin work on a vehicle from the queue of those awaiting repairs after breakdowns; and regardless of the size of the two queues, the vehicle just completing repair or service receives updated times for its next service event (breakdown or scheduled maintenance).

The list of factors, along with the low and high settings of interest, is provided below. X_1 and X_4 are integer-valued, the rest are continuous.

- X_1: Number of maintenance personnel, 2 to 8
- X_2: The ratio of the number of initial vehicles to the number of maintenance personnel: 5 to 10.
- X_3: Breakdown rate: (1 per 140 days) to (1 per 14 days)
- X_4: Maintenance cycle (days), 90 to 120
- X_5: Probability that standard maintenance suffices, 0.92 to 0.98.
- X_6: Probability that standard repair is required for a vehicle after a breakdown, 0.76 to 0.84
- X_7 and X_8: Two factors describing the parameters of the Weibull distribution for standard repair times. X_7 is the scale (α) that varies from 0.1 to 0.5, while X_8 is the shape (β) that varies from 1.5 to 5.

The standard service time is uniformly distributed between 5.5 and 6.5 hours. The standard repair time follows a Weibull distribution as parameterized by α and β. If a previously undiagnosed breakdown is identified during service (or repair), then t_{maint} (t_{repair}) follows a Weibull distribution with four times the mean of the standard repair time distribution.

The Ao model is stochastic, and a wide variety of performance measures can be calculated. We chose to examine the average vehicles available over a long period of time (Y) as a nearly-deterministic estimate of the steady-state mean vehicles available. To study the response surface of Y given the 8 input factors following the batch sequential method, an sFFLHD with batch size of 8 is used and GP models are fitted after each batch.

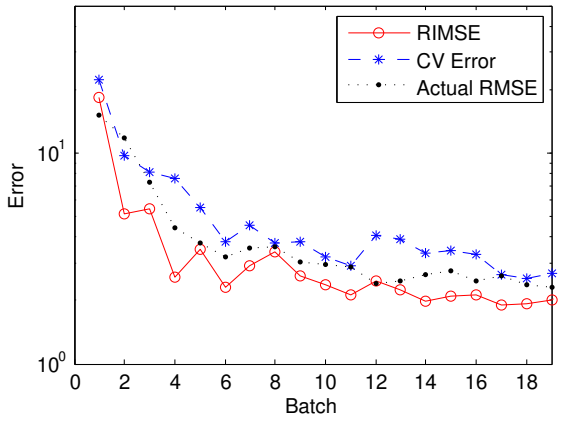

Figure 9: RIMSE, CV Error and Actual RMSE after Each Batch Stage (in Log Scale)

We start with a stopping criterion based on estimated $RIMSE$ from the GP models. With more batches of points evaluated, the fitted GP models tend to approximate the real response surface with smaller errors. However, improvement of fitting is not guaranteed after every batch stage. Figure 9 shows the $RIMSE$ after each batch stage. We choose to stop if the minimum $RIMSE$ from the five most recent batches is no more than a $p\%$ improvement over the minimum $RIMSE$ achieved in all batches before that. Specifically, we define $RIMSE1_b = min\{RIMSE_i, i = 1, \ldots, b-5\}$ and $RIMSE2_b = min\{RIMSE_i, i = b-4, \ldots, b\}$ for $b \geq 6$. The criterion stops the sequential experiments if ($RIMSE1 -$

$RIMSE2)/RIMSE1 < p\%$. We selected two possible scenarios ($p = 7.5$ and $p = 5$) and summarized the finding in Table 3. To assess the GP model fitting, actual model RMSEs were computed from a 10,000-point maximin LHD test points (see Figure 9).

Leave-one-out average cross validation error (CV error) could also be used to construct the stopping criterion. CV errors across batch stages are plotted in Figure 9. Similar to RIMSE criteria, we choose to stop if the minimum of the CV errors from the five most recent batches does not decrease by more than $p\%$ of the minimum CV error from batches before that. Similarly $CV1_b = min\{CV_i, i = 1, \ldots, b-5\}$ and $CV2_b = min\{CV_i, i = b-4, \ldots, b\}$ for $b \geq 6$. The criterion stops the sequential experiments if $(CV1 - CV2)/CV1 < p\%$. Results under $p = 7.5$ and $p = 5$ are shown in Table 3.

Batch(b)	RIMSE1	RIMSE2	CV1	CV2	RMSE
6	18.355	2.319 (87%)	22.365	3.787 (83%)	3.219
7	5.118	2.319 (55%)	9.678	3.787 (61%)	3.541
8	5.118	2.319 (55%)	8.111	3.756 (54%)	3.601
9	2.567	2.319 (10%)	7.552	3.756 (50%)	3.052
10	2.567	2.319 (10%)	5.530	3.214 (42%)	2.940
11	2.319	2.122 (9%)	3.787	2.911 (23%)	2.878
12	2.319	2.122 (9%)	3.787	2.911 (23%)	2.420
13	2.319	2.122 (9%)	3.756	2.911 (23%)	2.470
14	2.319	1.973 (15%)	3.756	2.911 (23%)	2.657
15	2.319	1.973 (15%)	3.214	2.911 (9%)	2.759
16	***2.122***	***1.973 (7%)***	***2.911***	***3.287 (0%)***	2.457
17	2.122	1.912 (10%)			2.618
18	2.122	1.912 (10%)			2.381
19	***1.973***	***1.912 (3%)***			

Table 3: Results from RIMSE and CV error stopping criteria

Table 3 shows that the batch sequential sampling stops at batch stage 16 and 19 (bolded) if RIMSE stopping criterion with $p = 7.5$ and $p = 5$ are used respectively. Both CV error stopping criteria stop the batch sequential sampling at batch stage 16. The actual MSE across the 10,000 test points from the GP model at batch stage 16 is only 1.7% of the total response surface variation. Both RIMSE and CV error stopping criterion were able to fit GP models with small errors using a reasonable number of design points. Comparing RIMSE with actual RMSE, the expected RMSEs from GP models are slightly smaller than

the true RMSEs as the GP models are approximations of the true response surface. The cross validation method tends to slightly overestimate the true RMSE.

Operational availability is a component of a Readiness and System Cost Trade-off Analysis and Management Tool under development for the U.S. Marine Corps. A surrogate model, such as the GP model described above, will allow program managers to use this tool interactively to assess the impact of critical acquisition and logistics decisions on both readiness and life cycle cost.

6. Conclusion

We have proposed a new batch sequential design sFFLHD. At any batch stage, sFFLHD is an LHD with uneven levels. At certain batch stages, sFFLHD achieves high levels of both projectivity and orthogonality by becoming orthogonal at the big and intermediate grid levels and becoming an LHD at the small grid level. To demonstrate its advantages, we have compared against various design methods in the context of estimating the mean and fitting a GP model to various test surfaces. In low dimensional examples, sFFLHD performs as well or nearly as well in terms of RMSEs of the GP model fit. In high dimensional examples, sFFLHD produces a fitted surface with lower RMSEs on average than any other sequential methods tested. For estimating the mean in a region, sFFLHD produces lower variances at stages where the design is an OALHD. Empirically, we have shown that sFFLHD dominates the other tested designs in two examples at all stages studied. In addition, we have examined another version of sFFLHD with a slight variation to determine whether it is important that each batch be an LHD at the big grid level. We found this property does contribute substantially to sFFLHD's good performance if the design does not reach the orthogonal stages. We also demonstrated the use of the method and some potential stopping criteria using a simulation for vehicle availability for a fleet of vehicles.

Appendix A. : Generating Non-overlapping OAs

Let $\mathbf{v}$ be a $1 \times L$ vector, $\mathbf{v} = [v_1, v_2, \ldots, v_D]$. The first two elements, v_1 and v_2 are set to be zero; the other elements can take values from $0, 1, 2, \ldots, L-1$. There are L^{D-2} unique vectors. Let A be the initial OA, $\mathbf{A}_{i.}$ be ith row with elements $a_{i1}, a_{i2}, \ldots, a_{iD}$, and B be the generated OA with $b_{ik} = (a_{ik} + v_k)\ mod\ L$. Within each orthogonal array, slices of $\mathbf{A}^r$, $\mathbf{A}^r_p$, are then randomized.

Prove B_1 and B_2 created by $\mathbf{v}_1$ and $\mathbf{v}_2$ ($\mathbf{v}_1 \neq \mathbf{v}_2$) do not share a same row, and $B_1, B_2, \ldots, B_{L^{D-2}}$ form a full factorial.

Proof. $b_{1ik} = (a_{ik} + v_{1k})\, mod\, L, b_{2ik} = (a_{ik} + v_{2k})\, mod\, L$. Therefore $b_{1ik} = b_2ik + (v_{2k} - v_{1k})\, mod\, L$. Let $\mathbf{v} = [v_1, v_2, \ldots, v_D]$ and $v_k = (v_{2k} - v_{1k})\, mod\, L$. B_1 is generated by B_2 given by $\mathbf{v}$. Each B_i has distinct L^2 rows which correspond to L^2 points in $[1, 2, \ldots, L]^D$. Therefore B_1 and B_2 do not share a same row and $B_1, B_2, \ldots, B_{L^{D-2}}$ form a full factorial design. □

Appendix B. : A subroutine, NB, for adding a new batch to sFFLHD (All matrices are updated: $\mathbf{X_b}$,$\mathbf{V_b}$,$\mathbf{W_b}$)

For subroutine NB, we denote $Q(l_b, j, b, k)$ as the set of available levels in small grid from $l_b(k-1)/L_b+1, l_bk/L+1, \ldots, l_b(k+1)/L_b$ for dimension j. k denotes a level from the intermediate grid, $k = 1, 2, \ldots, L_b, j = 1, 2, \ldots, D$. At any time, $Q(l_b, j, b, k) = \{l_b(k-1)/L_b + 1, l_bk/L+1, \ldots, l_b(k+1)/L_b\} \setminus \mathbf{V}_{\cdot j}$.

$NB(\mathbf{G}_{n_1:n_2,\cdot}, \epsilon_{n_1:n_2,\cdot}, b)$ is defined as:

For all i and j in $\mathbf{G}_{n_1:n_2,\cdot}$ and $\epsilon_{n_1:n_2,\cdot}$, $i = 1, \ldots, n_2 - n_1$, $j = 1, \ldots, D$

- Update $Q(l_b, j, b, G_{ij})$: $Q(l_b, j, b, G_{ij}) = \{l_b(G_{ij} - 1)/L_b + 1, l_bG_{ij}/L + 1, \ldots, l_b(G_{ij} + 1)/L_b\} \setminus V_{\cdot j}$

- Let $N = |Q(l_b, j, b, G_{ij})|$, $e_1 = \lceil \epsilon_{ij} N \rceil$, and $e_2 = e_1 - \epsilon_{ij} N$

- Choose e_1th number from the set $Q(l_b, j, b, G_{ij})$, denote by e

- Update the small grid design matrix by $v_{n_1+i-1,j} = e$, the intermediate grid design matrix by $m_{n_1+i-1,j} = G_{ij}$ and the big grid design matrix by $w_{n_1+i-1,j} = \lfloor LG_{ij}/L_b \rfloor$

- Sequential design matrix $x_{n_1+i-1,j} = (e - e_2)/l_b$

Next i, j

□

Example 7. Use Example 1 to demonstrate the subroutine NB

Suppose 3 batches of size 3 sequential designs have been drawn. $l_b = 27, b = 3$

$$\mathbf{G}_{10:12,\cdot} = \begin{bmatrix} 0 & 0 & 2 \\ 1 & 1 & 1 \\ 2 & 2 & 0 \end{bmatrix}, \epsilon_{10:12,\cdot} = \begin{bmatrix} 0.409 & 0.624 & 0.198 \\ 0.810 & 0.193 & 0.626 \\ 0.808 & 0.958 & 0.845 \end{bmatrix}.$$

For $i = 1, j = 1$, we have $Q(l_b, j, b, G_{ij}) = Q(27, 1, 3, 0) = \{1, 3, 4, 6, 7, 9\}$ and $\epsilon_{10,1} = 0.409$. Therefore,

$$N = 6, e_1 = 3, e_2 = 0.544.$$

The 3rd element of $Q(27, 1, 3, 0)$ is chosen

$$v_{10,1} = 4, m_{10,1} = 0, w_{10,1} = 0, x_{10,1} = (4 - 0.544)/27 = 0.128.$$

For other i and j, the algorithm follows similarly until we have

$$\mathbf{V}_{10:12,\cdot} = \begin{bmatrix} 4 & 6 & 21 \\ 16 & 12 & 15 \\ 26 & 27 & 8 \end{bmatrix}, \mathbf{M}_{10:12,\cdot} = \begin{bmatrix} 0 & 0 & 2 \\ 1 & 1 & 1 \\ 2 & 2 & 0 \end{bmatrix},$$

$$\mathbf{W}_{10:12,\cdot} = \begin{bmatrix} 0 & 0 & 2 \\ 1 & 1 & 1 \\ 2 & 2 & 0 \end{bmatrix}, \text{ and } \mathbf{X}_{10:12,\cdot} = \begin{bmatrix} 0.128 & 0.213 & 0.748 \\ 0.587 & 0.413 & 0.547 \\ 0.957 & 0.991 & 0.262 \end{bmatrix}.$$

Appendix C. : Proof of Proposition 2

Proof. It can be verified in the algorithm that each possible batch in the space is equally likely to be sampled. With each batch being an LHD, the expectation of the mean estimator from a single run is an unbiased estimator for the true average. Therefore, the mean estimator from the sequential design is unbiased. (2) is true.

Whenever the big grid design is an OA, it is of the form $OA(\lambda L^2, D, L, 2)$, $\lambda \in \mathbb{N}_+$. He and Qian (2011) has showed the variance structure as $L \to \infty$ when λ is fixed. (3) and (4) follow their proof.

To show (5), notice that when the intermediate grid design is a full factorial design with L_b levels, the variance of the mean estimator can be expressed as

$$Var(\overline{Y}) = \frac{1}{L_b^{2D}} \left[\sum_{i=1}^{L_b^D} Var(Y_i) + \sum_{i,j} Cov(Y_i, Y_j) \right].$$

The first part of the variance decomposition is the same as the lattice sampling variance under the uniform rectangle rule

$$X_{ij} = X_{ij}^c + u_{ij}/L_b, i = 1, \ldots, L_b^D, j = 1, \ldots, D$$

where X_{ij}^c's are the centers of the grids, u_{ij}'s are independent random uniform $(0, 1]$ numbers, and $Y_j = f(\mathbf{X}_{i.})$. Owen (1992) has shown the variance of lattice sampling is $O(L_b^{-D-2})$ under continuous f which is smaller than that of random sampling $O(L_b^{-D})$.

For i and j with $M_{im} \neq M_{jm}, \forall m \in 1, \ldots, D$, we have $Cov(Y_i, Y_j) = 0$.

For i and j, with $M_{ik} = M_{jk}$ for a particular dimension k and $M_{im} \neq M_{jm}, \forall m \in 1, \ldots, D, m \neq k$,

$$Cov(Y_i, Y_j) = E[(Y_i - \tau_i)(Y_j - \tau_j)] = E[(Y_i - \tau_i)E[(Y_j - \tau_j)|\mathbf{X}_{i.}]]$$

where τ_i and τ_j are $E(Y_i)$ and $E(Y_j)$ respectively.

$$E[(Y_j - \tau_j)|\mathbf{X}_{i.}] = -\frac{L_b}{l - L_b} E[(f(\mathbf{X}_{j.}) - \tau_j)|l_b \lfloor X_{jk} \rfloor = V_{ik}]$$

Here $l = L_b^D$. Given continuous f, $E[(f(\mathbf{X}_{j.}) - \tau_j)|l_b \lfloor X_{jk} \rfloor = V_{ik}] = O(L_b^{-1})$. Therefore $E[(Y_j - \tau_j)|\mathbf{X}_{i.}] = O(L_b^{-D})$ and $Cov(Y_i, Y_j) = E[(Y_i - \tau_i)E[(Y_j - \tau_j)|\mathbf{X}_{i.}]] = O(L_b^{-1})O(L_b^{-D}) = O(L_b^{-D-1})$

$$\frac{1}{L_b^{2D}} \sum_{R_1} Cov(Y_i, Y_j) = \frac{1}{L_b^{2D}} D(L_b - 1)^{D-1} L_b^D O(L_b^{-D-1}) = O(L_b^{-D-2})$$

where R_1 denotes the set of all i and j with $M_{ik} = M_{jk}$ for a particular dimension k and $M_{im} \neq M_{jm}, \forall m \in 1, \ldots, D, m \neq k$. Similarly for i and j with $M_{ik} = M_{jk}$ on more than one dimension, we found the covariance is $o(L_b^{-D-2})$. Therefore we have proven the variance of the mean estimator at FFLHD stage is $O(L_b^{-D-2})$. □

Appendix D. : RMSE Comparion of sFFLHD with bMmLHD, bLHD, and rsFFLHD in Examples 3-5 (A negative entry indicates better performance by sFFLHD and bold indicates significance at 0.05 level)

Borehole 4D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	**-2.2089**	-0.0223	**-0.0203**	**-0.0113**	**-0.0094**	**-0.0038**	**-0.0030**	**-0.0031**	**-0.0026**	**-0.0019**	**-0.0015**	**-0.0014**	-0.0003	-0.0006	-0.0002	-0.0004
sFFLHD-MmDist	0.2167	**-0.8769**	**-0.0239**	0.0042	0.0023	**-0.0018**	**-0.0057**	**-0.0062**	**-0.0059**	**-0.0042**	**-0.0046**	**-0.0033**	**-0.0028**	**-0.0026**	**-0.0018**	**-0.0020**
sFFLHD-rsFFLHD	**-3.2831**	-0.0872	**-0.0188**	-0.0038	**-0.0022**	-0.0005	-0.0007	-0.0004	-0.0001	0.0007	0.0007	0.0006	0.0007	0.0003	0.0006	0.0005
sFFLHD-bLHD	0.0985	0.0160	-0.0079	-0.0018	**-0.0061**	0.0006	-0.0012	**-0.0011**	**-0.0014**	-0.0008	0.0000	-0.0003	-0.0001	-0.0005	-0.0008	-0.0006

Borehole 6D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	**-4.8184**	**-1.3816**	**-0.8413**	**-0.1447**	-0.1934	**-0.0391**	**-0.0252**	**-0.0156**	**-0.0081**	**-0.0045**	**-0.0040**	**-0.0028**	**-0.0032**	**-0.0032**	**-0.0022**	**-0.0024**
sFFLHD-MmDist	-0.8138	-0.8442	**-2.2883**	**-0.6911**	**-0.1456**	-0.1029	-0.0019	0.0012	**-0.0065**	**-0.0357**	**-0.0326**	**-0.0211**	**-0.0107**	**-0.0036**	**-0.0018**	-0.0004
sFFLHD-rsFFLHD	**-3.8814**	-0.8242	-0.3037	**-0.0789**	**-0.0260**	-0.0051	0.0023	-0.0013	**-0.0039**	-0.0003	-0.0003	0.0003	-0.0005	**-0.0014**	**-0.0012**	-0.0008
sFFLHD-bLHD	-0.5474	-0.3844	-1.0507	-0.2215	-0.0377	-0.0180	-0.0170	-0.0147	-0.0066	-0.0025	-0.0022	-0.0012	-0.0019	-0.0023	-0.0019	-0.0021

Borehole 8D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	**-5.9585**	**-2.7276**	**-2.0030**	**-1.0016**	**-0.4959**	**-0.3984**	**-0.2773**	**-0.1952**	**-0.1015**	**-0.0388**	**-0.0392**	**-0.0416**	**-0.0361**	**-0.0162**	**-0.0164**	-0.0056
sFFLHD-MmDist	**-1.3142**	**-3.9382**	**-3.8971**	**-1.7045**	**-0.5692**	**-0.6005**	**-0.9263**	**-0.9275**	**-0.6835**	**-0.5807**	**-0.5225**	**-0.4756**	**-0.4634**	**-0.4399**	**-0.4160**	**-0.3643**
sFFLHD-rsFFLHD	**-5.7156**	**-1.3069**	-0.5506	0.0637	-0.0561	0.0750	-0.0779	**-0.0594**	**-0.0670**	0.0085	-0.0134	-0.0128	-0.0136	-0.0021	-0.0104	0.0022
sFFLHD-bLHD	**-1.4479**	**-1.7220**	**-1.4792**	**-0.6313**	**-0.2858**	**-0.2268**	**-0.2825**	**-0.1433**	**-0.0928**	**-0.0353**	**-0.0288**	-0.0030	**-0.0185**	-0.0070	**-0.0148**	-0.0080

theta=5

Gaussian 2D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	7.05E-03	**-6.45E-02**	-2.08E-02	-6.68E-03	**-4.93E-03**	**-4.96E-03**	**-3.28E-03**	**-1.66E-03**	**-1.10E-03**	**-4.86E-04**	**-3.39E-04**	**-2.29E-04**	**-6.91E-05**	**-5.18E-05**	**-3.85E-05**	**-2.88E-05**
sFFLHD-MmDist	8.58E-02	4.75E-02	3.84E-02	2.06E-02	7.30E-03	3.22E-03	1.47E-03	7.82E-04	5.56E-04	3.54E-04	2.39E-04	1.60E-04	8.93E-05	6.00E-05	3.64E-05	2.50E-05
sFFLHD-rsFFLHD	1.81E-02	**-3.15E-02**	-4.76E-03	2.73E-03	2.86E-04	4.85E-05	-1.55E-04	3.77E-05	-1.17E-04	-9.02E-05	-1.93E-05	3.62E-06	-1.72E-05	-6.81E-06	1.30E-06	3.67E-06
sFFLHD-bLHD	4.69E-02	4.06E-03	3.08E-03	-8.73E-04	**-2.74E-03**	**-2.71E-03**	**-2.34E-03**	**-1.43E-03**	**-5.60E-04**	**-4.44E-04**	**-2.84E-04**	**-1.26E-04**	**-1.02E-04**	**-4.78E-05**	**-3.90E-05**	**-3.95E-05**

Gaussian 4D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	-0.0310	-0.0243	-0.0068	-0.0215	**-0.0317**	-0.0197	**-0.0180**	**-0.0275**	**-0.0214**	**-0.0158**	**-0.0223**	**-0.0200**	**-0.0158**	**-0.0185**	**-0.0132**	-0.0075
sFFLHD-MmDist	-0.0139	-0.0448	-0.0089	-0.0318	-0.0084	-0.0236	-0.0135	0.0011	-0.0002	0.0127	0.0139	0.0218	0.0219	0.0182	0.0227	0.0244
sFFLHD-rsFFLHD	-0.0230	-0.0182	0.0157	0.0187	0.0005	-0.0085	-0.0047	-0.0009	-0.0089	-0.0109	**-0.0157**	-0.0072	-0.0041	-0.0045	-0.0016	0.0028
sFFLHD-bLHD	0.0150	-0.0110	0.0040	-0.0094	-0.0109	-0.0172	**-0.0168**	**-0.0219**	**-0.0172**	**-0.0134**	**-0.0181**	**-0.0112**	**-0.0094**	**-0.0139**	**-0.0147**	**-0.0147**

Gaussian 6D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	0.0092	-0.0008	-0.0086	-0.0133	-0.0116	-0.0070	0.0030	-0.0147	**-0.0166**	**-0.0159**	-0.0116	-0.0085	-0.0075	-0.0071	-0.0096	**-0.0142**
sFFLHD-MmDist	0.0132	0.0021	-0.0039	-0.0179	**-0.0439**	**-0.0461**	**-0.0439**	**-0.0651**	**-0.0669**	**-0.0823**	**-0.0703**	**-0.0535**	**-0.0568**	**-0.0462**	**-0.0497**	**-0.0432**
sFFLHD-rsFFLHD	0.0175	0.0016	0.0026	-0.0031	-0.0032	0.0058	0.0124	0.0100	-0.0025	-0.0026	-0.0041	0.0066	0.0062	0.0048	0.0038	0.0020
sFFLHD-bLHD	-0.0069	0.0023	0.0010	-0.0093	-0.0035	0.0056	0.0020	-0.0025	-0.0025	-0.0026	-0.0014	0.0031	0.0017	0.0012	-0.0013	-0.0042

Gaussian 8D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	-0.0077	-0.0072	-0.0051	-0.0026	-0.0003	0.0021	0.0063	0.0080	0.0016	0.0031	0.0009	-0.0006	0.0042	-0.0012	0.0064	0.0046
sFFLHD-MmDist	0.0054	-0.0021	-0.0122	-0.0093	-0.0095	-0.0119	**-0.0147**	**-0.0126**	**-0.0212**	**-0.0258**	**-0.0280**	**-0.0346**	**-0.0300**	**-0.0381**	**-0.0375**	**-0.0355**
sFFLHD-rsFFLHD	-0.0124	-0.0150	-0.0035	-0.0118	-0.0006	0.0005	0.0035	0.0055	-0.0039	-0.0033	0.0006	-0.0042	0.0003	-0.0033	0.0005	0.0007
sFFLHD-bLHD	-0.0081	-0.0130	-0.0093	-0.0070	-0.0003	-0.0031	0.0026	0.0033	-0.0009	-0.0009	-0.0005	-0.0003	0.0059	0.0003	0.0052	0.0020

theta=15

Gaussian 2D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	-0.0524	-0.0183	-0.0124	**-0.0475**	**-0.0349**	**-0.0335**	**-0.0251**	**-0.0227**	**-0.0134**	**-0.0121**	**-0.0106**	**-0.0063**	**-0.0043**	**-0.0030**	**-0.0016**	**-0.0022**
sFFLHD-MmDist	0.0111	0.0327	0.0845	0.0627	0.0397	0.0314	0.0186	0.0110	0.0139	0.0096	0.0080	0.0067	0.0054	0.0044	0.0038	0.0026
sFFLHD-rsFFLHD	-0.0474	0.0013	0.0170	-0.0202	**-0.0209**	**-0.0053**	**-0.0077**	-0.0009	0.0009	**-0.0023**	**-0.0031**	**-0.0025**	-0.0011	-0.0011	0.0001	0.0001
sFFLHD-bLHD	-0.0040	0.0153	0.0234	-0.0051	**-0.0223**	**-0.0169**	**-0.0275**	**-0.0255**	**-0.0158**	**-0.0170**	**-0.0095**	**-0.0068**	**-0.0053**	**-0.0045**	**-0.0034**	**-0.0023**

Gaussian 4D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	-0.0006	0.0153	0.0174	0.0034	-0.0015	-0.0030	0.0129	0.0138	0.0061	0.0085	-0.0068	-0.0047	-0.0074	-0.0098	**-0.0143**	**-0.0139**
sFFLHD-MmDist	-0.0192	-0.0119	0.0022	**-0.0314**	**-0.0355**	**-0.0391**	**-0.0319**	**-0.0399**	**-0.0551**	**-0.0497**	**-0.0614**	**-0.0474**	**-0.0447**	**-0.0344**	**-0.0373**	**-0.0240**
sFFLHD-rsFFLHD	0.0248	-0.0178	0.0093	-0.0050	0.0121	0.0023	0.0046	0.0099	-0.0024	0.0028	-0.0123	0.0022	-0.0035	-0.0032	-0.0037	-0.0010
sFFLHD-bLHD	0.0032	-0.0119	0.0174	-0.0100	0.0020	-0.0067	0.0004	-0.0009	-0.0005	0.0003	-0.0131	-0.0087	-0.0124	-0.0087	**-0.0143**	**-0.0167**

Gaussian 6D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	0.0058	0.0047	-0.0124	-0.0038	-0.0075	-0.0027	-0.0017	0.0020	0.0011	0.0004	0.0005	0.0013	-0.0029	-0.0020	0.0004	-0.0016
sFFLHD-MmDist	0.0014	0.0042	-0.0079	-0.0028	-0.0028	-0.0013	-0.0036	-0.0057	-0.0050	-0.0043	-0.0107	**-0.0133**	**-0.0173**	**-0.0185**	**-0.0191**	**-0.0189**
sFFLHD-rsFFLHD	0.0165	0.0076	-0.0067	-0.0033	-0.0013	0.0024	0.0019	-0.0004	-0.0003	0.0027	0.0015	-0.0011	-0.0010	-0.0008	-0.0008	-0.0041
sFFLHD-bLHD	-0.0003	-0.0006	**-0.0140**	-0.0087	-0.0048	0.0000	0.0015	0.0006	-0.0001	0.0025	0.0019	-0.0009	-0.0002	-0.0005	0.0029	-0.0003

Gaussian 8D

Batch	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sFFLHD-bMmLHD	0.0174	-0.0110	0.0110	0.0070	-0.0100	0.0016	**-0.0090**	**-0.0069**	**-0.0071**	**-0.0108**	**-0.0082**	**-0.0117**	**-0.0080**	**-0.0051**	**-0.0115**	-0.0048
sFFLHD-MmDist	-0.0003	-0.0266	0.0177	0.0020	**-0.0123**	**-0.0175**	**-0.0177**	**-0.0190**	**-0.0198**	**-0.0194**	**-0.0220**	**-0.0204**	**-0.0215**	**-0.0283**	**-0.0245**	**-0.0241**
sFFLHD-rsFFLHD	0.0034	**-0.0173**	0.0120	0.0098	0.0007	-0.0039	-0.0064	-0.0070	-0.0035	-0.0038	-0.0026	0.0000	-0.0018	-0.0018	-0.0053	0.0012
sFFLHD-bLHD	0.0047	-0.0044	0.0146	-0.0009	**-0.0150**	**-0.0050**	**-0.0174**	**-0.0053**	**-0.0045**	**-0.0054**	**-0.0115**	**-0.0078**	**-0.0059**	**-0.0041**	**-0.0034**	**-0.0032**

References

Ankenman, B., Nelson, B.L., Staum, J., 2010. Stochastic kriging for simulation metamodeling. Operations Research 58, 371–382.

Bernardo, M., Buck, R., Liu, L., Nazaret, W., Sacks, J., Welch, W., 1992. Integrated circuit design optimization using a sequential strategy. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 11, 361–372.

He, X., Qian, P.Z.G., 2011. Nested orthogonal array-based latin hypercube designs. Biometrika 98, 721–731.

Johnson, M., Moore, L., Ylvisaker, D., 1990. Minimax and maximin distance designs. Journal of Statistical Planning and Inference 26, 131 – 148.

Journel, A.G., H.C., 1978. Mining Geostatistics. Academic Press, San Diego, CA.

Lam, C., Notz, W., 2008. Sequential adaptive designs in computer experiments for response surface model fit .

Loeppky, J.L., Moore, L.M., Williams, B.J., 2010. Batch sequential designs for computer experiments. Journal of Statistical Planning and Inference 140, 1452 – 1464.

Matheron, G., 1963. Principles of geostatistics. Economic Geology 58, 1246–1266.

McKay, M.D., Beckman, R.J., Conover, W.J., 1979. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 21, pp. 239–245.

Morris, M.D., Mitchell, T.J., 1995. Exploratory designs for computational experiments. Journal of Statistical Planning and Inference 43, 381 – 402.

Owen, 1992. Orthogonal arrays for computer experiments, integration and visualization. Statistica Sinica Volume 2, Issue 2, pp 439–452.

Owen, A.B., 1994. Controlling correlations in latin hypercube samples. Journal of the American Statistical Association 89, pp. 1517–1522.

Qian, P.Z.G., Wu, C.F.J., 2009. Sliced space-filling designs. Biometrika 96, 945–956.

Ranjan, P., Bingham, D., Michailidis, G., 2008. Sequential experiment design for contour estimation from complex computer codes. Technometrics 50, 527–541.

Sanchez, P., 2006. As simple as possible, but no simpler: A gentle introduction to simulation modeling, in: Simulation Conference, 2006. WSC 06. Proceedings of the Winter, pp. 2–10.

Tang, B., 1993. Orthogonal array-based latin hypercubes. Journal of the American Statistical Association 88, pp. 1392–1397.

Worley, B., 1987. Deterministic uncertainty analysis. ORNL-6428 .