2017

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

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

Weitao Duan, Bruce E. Ankenman, Susan M. Sanchez & Paul J. Sanchez (2017) Sliced Full Factorial-Based Latin Hypercube Designs as a Framework for a Batch Sequential Design Algorithm, Technometrics, 59:1, 11-22.
http://hdl.handle.net/10945/58815

Supplementary materials for this article are available online. Please go to http://www.tandfonline.com/r/TECH

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

Weitao DUAN and Bruce E. ANKENMAN

Department of Industrial Engineering
and Management Science
Northwestern University
Evanston, IL 60208
(*WeitaoDuan2013@u.northwestern.edu*;
ankenman@northwestern.edu)

Susan M. SANCHEZ and Paul J. SANCHEZ

Department of Operations Research
Naval Postgraduate School
Monterey, CA 93943
(*smsanche@nps.edu*; *pjsanche@nps.edu*)

When fitting complex models, such as finite element or discrete event simulations, the experiment design should exhibit desirable 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 (OALHDs). 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. Supplementary materials are available online.

KEY WORDS: Computer experiments; Computer model; Metamodels; Simulation experiments; Space filling design.

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 (Germano et al. 1991) and stochastic simulations to optimize call center staffing (Aksin, Armony, and Mehrotra 2007). Although the power and speed of computers have increased dramatically during the last few decades, a single evaluation of some computer models can still take hours or even days. If the computer models are computationally expensive, metamodels, sometimes referred to as surrogate models, can be constructed to approximate the complex computer models with sufficient accuracy. These metamodels can then replace the original computer models in optimization or "what if" analyses.

Building a metamodel for a computer simulation involves sampling a set of points from the design space and fitting a model to the observed data. The focus of this article is on design of experiments, that is, selecting the set of points to sample from the design space. We presume that kriging (or Gaussian process modeling) will be used for the fitted model. Kriging, developed in geostatistics (Matheron 1963; Journel and Huijbregts 1978), assumes spatial correlation between points. Responses at unobserved points are predicted using correlations between the observed points to create a response surface model. Kriging has become widely used for building metamodels of complex deterministic computer experiments, and Ankenman, Nelson, and Staum (2010) recently 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 little is known about the true underlying response surface.

A variety of experiment designs have been presented in the literature for supporting kriging models. When the goal of the metamodel is to fully map the region of interest, designs use space-filling criteria and seek to place points in the design space uniformly. McKay, Beckman, and Conover (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 (1992) 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 by OA(n, m, L, t). For every t columns, the L^t level combinations appear the same number of times. To construct an OALHD from an OA$(L^2, m, L, 2)$, 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 values are then randomly shuffled within each group, and each entry in the first column of the OA is replaced by the next available value from its corresponding group. The values within the groups are randomly reshuffled before replacing the entries of the next column

TECHNOMETRICS, FEBRUARY 2017, VOL. 59, NO. 1
DOI: 10.1080/00401706.2015.1108233
Color versions of one or more of the figures in the article can be found online at *www.tandfonline.com/r/tech*.

in the OA. OALHDs have good projectivity in any univariate and bivariate subspace if strength 2 OAs are used in construction. He and Ai (2011) proposed a new class of Latin hypercube designs with higher-dimensional uniformity when projected onto the columns corresponding to higher strength orthogonal arrays, as well as two-dimensional projective uniformity.

Other space-filling criteria have also been adopted when constructing designs. Johnson, Moore, and Ylvisaker (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 that 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. Cioppa and Lucas (2007) constructed nearly orthogonal Latin hypercube (NOLH) designs by combining correlation and distance performance measures. Related approaches include the multi-objective optimization approach of Joseph and Hung (2008), and the mixed integer programming approach of Hernandez, Lucas, and Carlyle (2012). Ranjan and Spencer (2014) presented a class of Latin hypercube designs based on nearly OAs.

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, Bingham, and Michailidis (2008) presented sequential designs with the objective of contour estimation. Lam (2008) proposed sampling additional points that maximize the expected improvement in model fit. Distance-based criteria also apply to the construction of sequential designs. Besides maximin and minimax criteria, Johnson, Moore, and Ylvisaker (1990) examined a weighted distance criterion for choosing new design points.

Recently, Loeppky, Moore, and Williams (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 (similar to the partitions of OALHD) within that bin structure are used to construct augmenting sets of runs that yield, as nearly as possible, aggregate designs that have maximin distance with near Latin hypercube sampling (LHS) at each batch stage. A batch sequential experiment 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 article, 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, Moore, and Williams (2010). Like Loeppky, Moore, and Williams's (2010) 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 sequential process, which we call the *golden stages*, our design becomes what we call a sliced full factorial-based Latin hypercube design (sFFLHD) that has very special space filling properties.

The remainder of this article is organized as follows. In Section 2, we define the sFFLHD and discuss its characteristics. In Section 3, we present a method for sequentially building an sFFLHD, one slice at a time. If each slice is observed as it is created, this construction method becomes our proposed batch sequential experiment design. In Section 4, we show how to continue beyond the first sFFLHD to sequentially create additional sFFLHDs so that the batch sequential design can continue indefinitely. We derive some theoretical properties of sFFLHDs in Section 5. In Section 6, we compare the results obtained using different design procedures for several numeric examples, and propose some choices of stopping criteria. In Section 7, we demonstrate an application of sFFLHD to a logistics simulation model. We summarize our work and present our conclusions in Section 8.

2. SLICED FULL FACTORIAL-BASED LATIN HYPERCUBE DESIGN

We now define a sliced full factorial-based Latin hypercube design (sFFLHD), which our sequential design achieves at the golden stages. A D-dimensional L^D-point design $\mathbf{X}$ 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 $n = L^D$ equally spaced levels given by $(0, 1/n], (1/n, 2/n], \ldots, ((n-1)/n, 1]$. This design can be sliced in the same sense that Qian and Wu (2009) sliced orthogonal arrays. An sFFLHD is an FFLHD that has been divided into slices of equal size, each of which forms an LHD when partitioned into L levels.

Figure 1 shows an example with $L = 3$, $D = 2$, and $n = 9$. The design denoted $\mathbf{V}$ is an LHD because it has all nine levels represented in each dimension. If this design is partitioned into three levels by mapping $\{1, 2, 3\}$ to $\{0\}$, $\{4, 5, 6\}$ to $\{1\}$, and $\{7, 8, 9\}$ to $\{2\}$, then $\mathbf{V}$ becomes the sliced full factorial design $\mathbf{W}$ in Figure 1. Alternatively, given a nine-run sliceable full factorial $\mathbf{W}$, we can construct a $\mathbf{V}$ that has this mapping property using a forward substitution mechanism. In the first column of $\mathbf{W}$, replace the three entries of 0 with a random permutation of the integers $\{1, 2, 3\}$, replace the three entries of 1 with a random permutation of the integers $\{4, 5, 6\}$, and replace the three entries of 2 with a random permutation of the integers $\{7, 8, 9\}$. Use separate permutations to populate column 2 of $\mathbf{V}$.

We call $\mathbf{W}$ the big grid design, and $\mathbf{V}$ the small grid design, because more levels result in smaller grid squares. For continuous-valued factors, the entries of $\mathbf{V}$ can be used as upper

$$\mathbf{W} = \left[\begin{array}{cc} 0 & 2 \\ 1 & 0 \\ 2 & 1 \\ \hdashline 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ \hdashline 0 & 1 \\ 1 & 2 \\ 2 & 0 \end{array}\right] \qquad \mathbf{V} = \left[\begin{array}{cc} 2 & 7 \\ 5 & 2 \\ 9 & 5 \\ \hdashline 1 & 3 \\ 4 & 6 \\ 8 & 9 \\ \hdashline 3 & 4 \\ 6 & 8 \\ 7 & 1 \end{array}\right]$$

Figure 1. A nine-run sliced full factorial design matrix $\mathbf{W}$, and a nine-run LHD $\mathbf{V}$ that maps to $\mathbf{W}$ when partitioned into three levels by mapping $\{1, 2, 3\}$ to $\{0\}$, $\{4, 5, 6\}$ to $\{1\}$, and $\{7, 8, 9\}$ to $\{2\}$.

endpoints of unit length intervals. For example, the entry 3 can represent the upper endpoint of the interval $(2, 3]$. When scaled to the unit cube, the entries in each dimension mark out equally spaced regions between 0 and 1. Using this framework, the design matrix, $\mathbf{X}$, can be simply a scaled version of the small grid design, such that $\mathbf{X} = \mathbf{V}/L^D$: we show this as $\mathbf{X}_{\text{scaled}}$ in Figure 2. Alternatively, the design matrix can be based on LHS, where the design point is randomly chosen within the interval specified by the entry in the small grid design: we show an example as $\mathbf{X}$ in Figure 2. Both $\mathbf{X}_{\text{scaled}}$ and $\mathbf{X}$ in Figure 2 meet the conditions for sFFLHD designs. We can also apply maximin distance sampling, where the design point is chosen within the interval specified by the entry in $\mathbf{V}$ to maximize the minimum distance between all points. We have found that maximin distance sampling performs better than LHS in terms of root mean squared error (RMSE) fitting, especially in lower dimensional empirical examples. The examples in Section 6 use the maximin distance criterion. Supplementary materials are available online.

3. SEQUENTIAL CONSTRUCTION OF AN sFFLHD

At a high level, our algorithm observes batches of L design points (i.e., slices from an sFFLHD) sequentially until a stopping criterion is reached or an L-level L^D-point sFFLHD is constructed. If we do reach the sFFLHD design, we call this a golden stage since the design is now a full factorial on the big grid scale and is an LHD on the small grid scale.

To sequentially construct an sFFLHD, we consider a special type of orthogonal array $\text{OA}(n, m, L, t)$, $t = 2$, $n = L^2$. Since the OA is of strength 2, then for any two columns, all level combinations appear exactly once. The $\text{OA}(9, 4, 3, 2)$ in Figure 3 is an example. If this OA is sorted by the first column, it can be sliced into three slices of three rows each as in Figure 3. Columns 2–4 of each slice form different Latin

$$\mathbf{X}_{scaled} = \mathbf{V}/9 = \left[\begin{array}{cc} 0.222 & 0.778 \\ 0.556 & 0.222 \\ 1.000 & 0.556 \\ \hdashline 0.111 & 0.333 \\ 0.444 & 0.667 \\ 0.889 & 1.000 \\ \hdashline 0.333 & 0.444 \\ 0.667 & 0.889 \\ 0.778 & 0.111 \end{array}\right] \qquad \mathbf{X} = \left[\begin{array}{cc} 0.183 & 0.638 \\ 0.479 & 0.122 \\ 0.856 & 0.428 \\ \hdashline 0.095 & 0.251 \\ 0.386 & 0.564 \\ 0.731 & 0.816 \\ \hdashline 0.203 & 0.322 \\ 0.596 & 0.713 \\ 0.605 & 0.056 \end{array}\right]$$

Figure 2. Two sFFLHD designs constructed from $\mathbf{V}$: $\mathbf{X}_{\text{scaled}}$ (using division) and $\mathbf{X}$ (using LHS).

$$OA(9,4,3,2) = \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 2 \\ 0 & 2 & 2 & 1 \\ \hdashline 1 & 0 & 2 & 2 \\ 1 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \\ \hdashline 2 & 0 & 1 & 1 \\ 2 & 1 & 2 & 0 \\ 2 & 2 & 0 & 2 \end{array}\right] = \left[\begin{array}{c|c} 0 & \\ 0 & \\ 0 & \\ 1 & \\ 1 & \mathbf{W}^1 \\ 1 & \\ 2 & \\ 2 & \\ 2 & \end{array}\right].$$

Figure 3. Construction of an initial, sliceable orthogonal array $\mathbf{W}^1$ from an $\text{OA}(9, 4, 3, 2)$.

hypercubes and columns 2–4 form an OA that we call $\mathbf{W}^1$. In fact, an $\text{OA}(L^2, D + 1, L, 2)$ can always be sliced into L slices of D-dimensional Latin hypercubes by using one column to separate the slices and then removing the column used for slicing. If using an OA with more than $D + 1$ columns, use the first column for slicing, and keep only D columns. The batches (slices) of the sFFLHD design are constructed using a series of D-factor orthogonal arrays $\mathbf{W}^1, \mathbf{W}^2, \ldots, \mathbf{W}^{L^{D-2}}$ with strength t ($t \geq 2$), each with L levels labeled $\{0, 1, 2, \ldots, L - 1\}$. Each OA is sliced into L slices as shown above. The OAs are nonoverlapping fractions of a full factorial design with L levels and D factors. By nonoverlapping, we mean that no two of the OAs contain the same row. In Appendix B in the online supplement, we show how to create $L^{D-2} - 1$ nonoverlapping OAs, $\mathbf{W}^2, \ldots, \mathbf{W}^{L^{D-2}}$, from $\mathbf{W}^1$. Each new $\mathbf{W}^i$ is constructed from $\mathbf{W}^1$ using a carefully chosen $1 \times D$ vector $\mathbf{v}^i$.

In the process of batch sequential sampling, four designs are created: the big grid design, the intermediate grid design, the small grid design, and the actual design matrix. The big grid design, $\mathbf{W}$, builds orthogonality in L levels and is constructed sequentially from the batches of the $\mathbf{W}^i$'s. It achieves orthogonality each time the number of observations, n, is a multiple of L^2. The small grid design, $\mathbf{V}$, has integer entries that are all greater than or equal to one. It builds one-dimensional projectivity in each dimension, and becomes an LHD each time n is equal to a power of L. Initially, entries in $\mathbf{V}$ take on values $v \in \{1, 2, \ldots, L^2\}$; each time n is equal to L^c for some integer $c > 1$, the upper limit is rescaled to L^{c+1}. In this way, we build Latin hypercube projectivity on more than L^2 levels as sampling progresses. At the first golden stage, the small grid design becomes an L^D-level LHD. The intermediate grid design, $\mathbf{M}$, begins as the big grid design but comes into play after we have reached a golden stage; it builds orthogonality in rL levels for some integer $r \geq 1$. The design matrix, $\mathbf{X}$, is a scaled version of the small grid design that fits inside the D-dimensional unit cube. Notationally, $\mathbf{W}_{i:j}$ represents batches i through j, so $\mathbf{W}_{b:b}$ represents the bth batch (similarly for $\mathbf{M}$, $\mathbf{V}$, and $\mathbf{X}$). The example in Figure 4, where $D = L = 3$, illustrates the design matrices for the first three batches when LHS is used for $\mathbf{X}$.

The fourth batch for the big grid design is the first slice of the second $\text{OA}((9, 4, 3, 2)$, $\mathbf{W}^2$. Since $n = 9 = L^2$, we must rescale $\mathbf{V}$ to L^3 levels before proceeding. The first three batches take on the rescaled values $\mathbf{V}_{1:3} = \lceil L^3 \mathbf{X}_{1:3} \rceil$: this ensures that the previously constructed design points remain aligned with the rescaled small grid design. Let "\" be a set exclusion operator, and let v_{ij} and w_{ij} denote the values in the ith row and jth column of $\mathbf{V}_{1:4}$ and $\mathbf{W}_{1:4}$, respectively.

Run	Batch	
1		
2	1	
3		
4		
5	2	$\mathbf{W}^1$
6		
7		
8	3	
9		

$$\mathbf{W}_{1:3} = \mathbf{M}_{1:3} = \left[\begin{array}{ccc} 0&0&0\\1&1&2\\2&2&1\\ \hdashline 0&2&2\\1&0&1\\2&1&0\\ \hdashline 0&1&1\\1&2&0\\2&0&2 \end{array}\right], \mathbf{V}_{1:3} = \left[\begin{array}{ccc} 3&2&2\\4&4&7\\7&9&4\\ \hdashline 2&8&8\\6&3&6\\9&6&1\\ \hdashline 1&5&5\\5&7&3\\8&1&9 \end{array}\right], \mathbf{X}_{1:3} = \left[\begin{array}{ccc} 0.281&0.172&0.193\\0.343&0.400&0.695\\0.681&0.953&0.403\\ \hdashline 0.149&0.885&0.878\\0.648&0.303&0.575\\0.922&0.662&0.034\\ \hdashline 0.062&0.472&0.483\\0.484&0.747&0.315\\0.810&0.036&0.987 \end{array}\right]$$

Figure 4. On reaching three slices when $D = L = 3$: $\mathbf{W}_{1:3} = \mathbf{M}_{1:3}$, the big and intermediate grid designs; $\mathbf{V}_{1:3}$, the small grid design (before rescaling); and the design matrix $\mathbf{X}_{1:3}$ from $\mathbf{V}_{1:3}$ with LHS.

For the new batch, corresponding to $i = 10, 11, 12$, v_{ij} is randomly selected from the integers $\{Lw_{ij} + 1, \ldots, Lw_{ij} + L^2\} \setminus \{v_{hj} | h < i, w_{hj} = w_{ij}\}$. In Figure 5, $v_{10,2}$ (corresponding to $w_{10,2} = 0$) was randomly drawn from the set $\{1, 2, \ldots, 9\} \setminus \{v_{12}, v_{52}, v_{92}\} = \{2, 3, 4, 6, 7, 8\}$, and the remaining v_{ij} for the new batch are selected in a similar manner. Finally, we use the new slice of $\mathbf{V}$ to obtain the new slice of $\mathbf{X}$ via LHS.

Batches continue to be created sequentially in this manner, rescaling $\mathbf{V}$ each time we move past a batch where n is an integer power of L, until a golden stage is reached. By construction, whenever the number of batches $b = cL^{D-1}$ for some integer c, the current design can be considered to yield c replicates of the initial big grid design, $\mathbf{W}_{1:L^{D-1}}$.

Figure 6 illustrates the early stages of the procedure when $D = L = 3$. Solid black lines represent the big grid (three levels), dashed black lines in subplots (b) and (c) represent the small grid through the first three batches, before rescaling (nine levels), and gray lines in (c) represent the rescaled small grid at the first golden stage (27 levels). Batch numbers are provided in subplot (a), which shows that each of the first three batches is a three-level LHD, and together they form a three-level OA on the big grid. Subplot (b) shows the same points, with shaded regions emphasizing how they form an LHD on the nine-level grid. Subplot (c) shows the design at the first golden stage; there are three points in each of the nine big-grid squares because this is a two-dimensional projection of a 3^3 factorial on the big grid—and at the same time, the design is a 27-level LHD as shown by the gray lines.

4. SEQUENTIAL CONSTRUCTION OF ADDITIONAL sFFLHDS

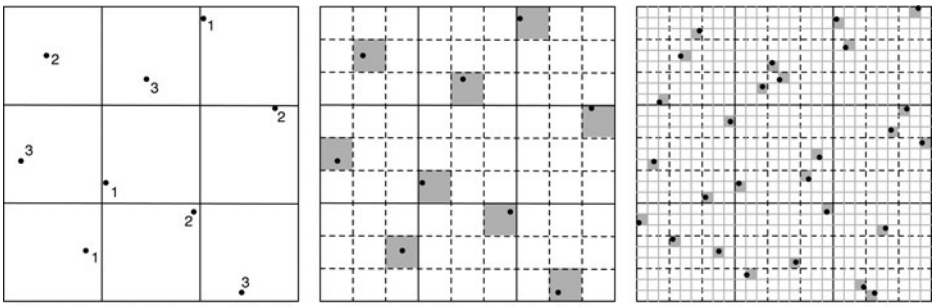
After the first golden stage, we seek to build orthogonality on more than L levels and Latin hypercube projectivity on more than L^D levels. To build LHD projectivity on more than L^D levels, the small grid design gets rescaled whenever $n = L^c$ for some integer c as previously described. Similarly, $\mathbf{X}$ can be used to rescale the intermediate grid design, $\mathbf{M}$, after each new golden stage. Before moving past a golden stage, we must choose a rescaling integer r that satisfies the condition $r^q = L$ for some $q \in \mathbb{N}_+$. In this article, we assume that r is as small as possible to allow the design to reach each subsequent golden stage as quickly as possible. Sometimes $r = L$ is the only possibility. We then rescale the existing intermediate grid design $\mathbf{M}$ to rL levels as follows: $\mathbf{M}_{1:L^D} = \lceil (rL)\mathbf{X}_{1:L^D} \rceil$. Continuing our example, Figure 7 shows the four designs when the first golden stage is reached (before rescaling). Using $r = 3$, Figure 8 shows the small grid design rescaled to 81 levels and the intermediate grid design rescaled to 9 levels, in preparation for further sampling. Figure 8 also provides the intermediate and small grid designs for batches 10 through 18.

Note that the first golden stage is reached at batch $b = L^{D-1}$ (i.e., $n = L^D$). Our goal for each subsequent set of L^{D-1} batches of the intermediate grid design is that the set is simultaneously an L-level L^D-point sFFLHD, and an rL-level L^D-point fractional factorial that does not overlap with previously constructed sFFLHDs when projected on the rescaled small and intermediate grids. The procedure in Appendix C in the online supplement shows how to construct these nonoverlapping fractional factorials. After r sets of L^{D-1} batches have been observed, rL^D levels will have been used in each column of the small grid design. This creates an LHD in rL^D levels, and the intermediate grid design will be a full factorial in rL levels. Thus, we reach the second golden stage, where $\mathbf{X}_{1:(rL)^D}$ is an rL-level $(rL)^D$-point sFFLHD. We then rescale the intermediate grid design to r^2L levels and the process continues. In this manner, we continue to build toward an r^2L-level $(r^2L)^D$-point sFFLHD at the next golden stage.

Run	Batch
1	
2	1
3	
4	
5	2
6	
7	
8	3
9	
10	
11	4
12	

$$\mathbf{W}_{1:4} = \mathbf{M}_{1:4} = \left[\begin{array}{ccc} 0&0&0\\1&1&2\\2&2&1\\ \hdashline 0&2&2\\1&0&1\\2&1&0\\ \hdashline 0&1&1\\1&2&0\\2&0&2\\ \hdashline 0&0&2\\1&1&1\\2&2&0 \end{array}\right], \mathbf{V}_{1:4} = \left[\begin{array}{ccc} 8&5&6\\10&11&19\\19&26&11\\ \hdashline 5&24&24\\18&9&16\\25&18&1\\ \hdashline 2&13&14\\14&21&9\\22&1&27\\ \hdashline 4&6&21\\16&12&15\\26&27&8 \end{array}\right], \mathbf{X}_{1:4} = \left[\begin{array}{ccc} 0.281&0.172&0.193\\0.343&0.400&0.695\\0.681&0.953&0.403\\ \hdashline 0.149&0.885&0.878\\0.648&0.303&0.575\\0.922&0.662&0.034\\ \hdashline 0.062&0.472&0.483\\0.484&0.747&0.315\\0.810&0.036&0.987\\ \hdashline 0.128&0.213&0.748\\0.587&0.413&0.547\\0.957&0.991&0.262 \end{array}\right]$$

Figure 5. On reaching four slices when $D = L = 3$: $\mathbf{W}_{1:4} = \mathbf{M}_{1:4}$, the big and intermediate grid designs; $\mathbf{V}_{1:4}$, the small grid design (rescaled after three slices); and the design matrix $\mathbf{X}_{1:4}$, where the first three slices remain unchanged, and the fourth slice comes from $\mathbf{V}_{1:4}$ with LHS.



(a) Factor 2 vs Factor 1, three-level big grid (b) Factor 2 vs Factor 1, nine-level grid (c) Factor 2 vs Factor 1, 27-level grid

Figure 6. Two-dimensional projections of Factor 2 versus Factor 1, where (a) shows the results from $\mathbf{X}_{1:3}$ on the big grid, with numbers indicating the batch, (b) shows $\mathbf{X}_{1:3}$ one a nine-level small grid, and (c) shows the results at the first golden stage, $\mathbf{X}_{1:9}$, on a 27-level grid.

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

Figure 7. On reaching the first golden stage, $\mathbf{X}_{1:9}$, an sFFLHD when $D = L = 3$ with LHS; $\mathbf{W}_{1:9}$, the associated big grid design; $\mathbf{M}_{1:9}$, the associated intermediate design (before rescaling); and $\mathbf{V}_{1:9}$, the associated small grid design (before rescaling).

Batch	rescaled after first nine batches: $\mathbf{M}_{1:9}=$	$\mathbf{V}_{1:9}=$	Batch	second nine batches: $\mathbf{M}_{10:18}=$	$\mathbf{V}_{10:18}=$
1	2 1 1	23 14 16	10	2 2 0	20 19 1
	3 3 6	28 33 57		3 4 8	29 41 74
	6 8 3	56 78 33		6 6 5	55 60 50
2	1 7 7	13 72 72	11	1 8 6	10 73 60
	5 2 5	53 25 47		5 0 4	46 2 37
	8 5 0	75 54 3		8 3 2	73 30 20
3	0 4 4	5 39 40	12	0 5 3	2 48 28
	4 6 2	40 61 26		4 7 1	37 64 13
	7 0 8	66 3 80		7 1 7	70 10 65
4	1 1 6	11 18 61	13	1 2 8	15 23 73
	5 3 4	48 34 45		5 4 3	47 37 30
	8 8 2	78 81 22		8 6 1	80 56 10
5	0 6 3	7 55 36	14	0 7 5	3 68 46
	4 1 0	45 11 9		4 2 2	39 20 19
	8 4 7	79 44 68		8 5 6	81 46 55
6	2 5 0	27 50 4	15	2 3 2	21 31 23
	4 7 6	38 66 58		4 8 8	44 75 81
	6 0 5	63 5 53		6 1 4	50 12 41
7	0 2 3	1 22 29	16	0 0 5	6 1 48
	5 4 2	51 40 21		5 5 1	50 49 11
	6 7 7	58 69 64		6 8 6	57 80 56
8	1 8 1	18 74 12	17	1 6 0	16 57 4
	3 0 8	31 8 75		3 1 7	30 13 67
	7 5 5	71 47 51		7 3 4	64 28 38
9	2 3 8	19 28 77	18	2 4 7	24 38 66
	3 6 4	35 59 39		3 7 3	32 71 31
	7 2 1	69 21 15		7 0 0	65 9 2

Figure 8. Rescaled design matrices after the first golden stage with $D = L = 3$: $\mathbf{M}_{1:9}$, the rescaled intermediate grid design with $rL = 9$ levels; and $\mathbf{V}_{1:9}$, the rescaled small grid design with $rL^D = 81$ levels. New $\mathbf{M}$ and $\mathbf{V}$ for the nine batches immediately following this golden stage are also provided.

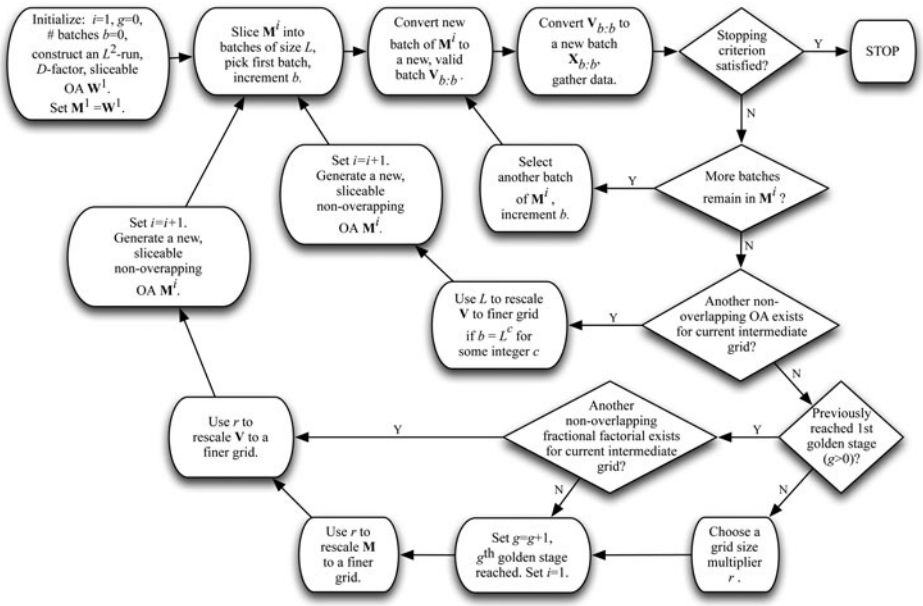


Figure 9. Flowchart of sFFLHD construction.

The small grid design for the next set of L^{D-1} batches is a reverse mapping of the intermediate grid design. Each entry of $\{i\}$ in the intermediate grid design is mapped to one of the integers from $iL^{D-1}+1$ to $(i+1)L^{D-1}$ in the small grid design. In the example in Figure 8, $\{0\}$ is mapped to $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, $\{1\}$ is mapped to $\{10, 11, 12, 13, 14, 15, 16, 17, 18\}$, and so forth. To ensure LHD projectivity in the small grid design, the algorithm must, on a column-by-column basis, avoid reusing levels from earlier batches of the rescaled $\mathbf{V}$.

Figure 9 shows a flowchart of sFFLHD construction. Essentially, the inner loop (upper right corner) sequentially generates batches from a sliceable OA. The middle loop generates nonoverlapping OAs from the initial OA, and builds LH projectivity on successively finer grids. The outer loop sequentially generates new OAs (hence building orthogonality) on successively finer grids, as additional golden stages are reached. The procedure terminates whenever the stopping criterion is satisfied. The stopping criterion can be based either on the design (e.g., maximum desired number of batches is reached), or on characteristics of the response.

5. 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 both accuracy and precision. Space filling designs can be used for many different purposes. While our main focus is kriging, another common application is mean estimation over a multi-dimensional space. The variance of the mean estimator can be used as a model-independent criterion for judging the quality of space filling designs (Qian and Wu 2009). We now show that the mean estimator has lower variance from an sFFLHD than from random sampling, especially at certain stages.

5.1 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 $\int_{(0,1]^D} f(\mathbf{x})^2 dF < \infty$ 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 $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$. The sample mean $\overline{Y}$ of n runs, $\overline{Y} = \frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i)$, is used as a predictor for μ. We will now study the quality of $\overline{Y}$ for an n-run sFFLHD.

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 analysis of variance (ANOVA) decomposition of f (Owen 1994) is given by $f = \sum_{u\in\mathcal{D}} \alpha_u$, where 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 $\sigma^2 = \int (f-\mu)^2 dF$ is simply $\sum_{u\in\mathcal{D}\setminus\varnothing} \int \alpha_u^2 dF$.

Appendix D in the online supplement establishes the forms of the marginal and joint probability mass functions. Following the derivation in Owen (1994) with some slight changes in notation, let w_{ij} denote the jth entry of the ith row of the big grid design $\mathbf{W}$. For $u \subset \mathcal{D}$, let $\eta_{ij}(u) = \{k \in u : w_{ik} = w_{jk}\}$ and define

$$S(u, r) = \sum_{i=1}^n \sum_{j=1}^n 1\{|\eta_{ij}(u)| = r\}$$

and for batch b,

$$S_b(u, r) = \sum_{i=(b-1)L+1}^{bL} \sum_{j=(b-1)L+1}^{bL} 1\{|\eta_{ij}(u)| = r\}.$$

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

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

Using the probability mass functions in Appendix D in the online supplement, we can derive the expectation and variance of the mean estimator of sFFLHD. We use $\mathbf{x}_{i.}$ to represent the ith row of the small grid design, $\mathbf{X}$.

Proposition 1. Let $\overline{Y_b} = \frac{1}{L} \sum_{i=(b-1)L+1}^{bL} f(\mathbf{x}_{i.})$, the mean estimator using a single batch of sFFLHD. Then

$$E(\overline{Y_b}) = \mu \quad \text{and} \quad E(\overline{Y}) = \mu. \tag{1}$$

For a continuous function f, as $L \to \infty$

$$\text{var}(\overline{Y_b}) = \sum_{|u|\geq 2} S_b(u, |u|) L^{-2} \text{var}(\alpha_u(\mathbf{x})) + o(L^{-1}). \tag{2}$$

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

$$\text{var}(\overline{Y}) = \sum_{|u|\geq 3} S(u, |u|) n^{-2} \text{var}(\alpha_u(\mathbf{x})) + o(n^{-1}). \tag{3}$$

Let L_b be the number of levels of $\mathbf{M}$. At stages where $n = L_b^D$, the sequential design is an FFLHD, and as $n \to \infty$ we have

$$\text{var}(\overline{Y}) = O(L_b^{-D-2}). \tag{4}$$

The proof appears in Appendix A in the online supplement.

From Proposition 1, (1) shows that the mean estimator of each batch and the whole sequential design at any batch stage is unbiased. We know from Tang (1993) that the variance achieved by an ordinary Latin hypercube design under continuous f is $o(L^{-1})$, which is lower than the $O(L^{-1})$ variance of random sampling. (2) shows that the variance achieved by each batch of our procedure is similar to that of an ordinary LHD. (3) shows that variance differences are more pronounced when the sequential design is an OALHD, as pointed out in He and Qian (2011). (4) shows that when the sequential design becomes an FFLHD, the variance is $O(L_b^{-D-2})$, which is similar to lattice sampling (Owen 1992). With the structure of the three grid designs, we attain good sampling properties even at these intermediate stages. We demonstrate this empirically in the next section.

6. EMPIRICAL EXAMPLES

For Examples 1 and 2, we sample eight points at a time, setting the final budget to 16 batches (128 runs). For each design method, we scale all designs to fit the range of interest, generate 100 independent designs, evaluate the response functions at the design points, and fit a GP model at each batch stage. We use a 10,000 point maximin LHD to assess the RMSE of each GP model. In Section 6.4, we study the variances of the mean estimators.

6.1 Comparison of sFFLHD and MmDist

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

Example 1. Borehole Example

Worley (1987) used a model to demonstrate the flow of water through a borehole. The model has eight input variables. In our comparison, we vary four, six, and all of the eight variables. 95% confidence intervals of RMSE differences are obtained via the 100 independent replications. Since the difference is (RMSE for sFFLHD)−(RMSE for MmDist), a 95% confidence interval that is strictly negative indicates that sFFLHD performs better. Because the ranges of RMSE differences vary substantially across the batches, batches 1–6 are shown in the first row of Figure 10,

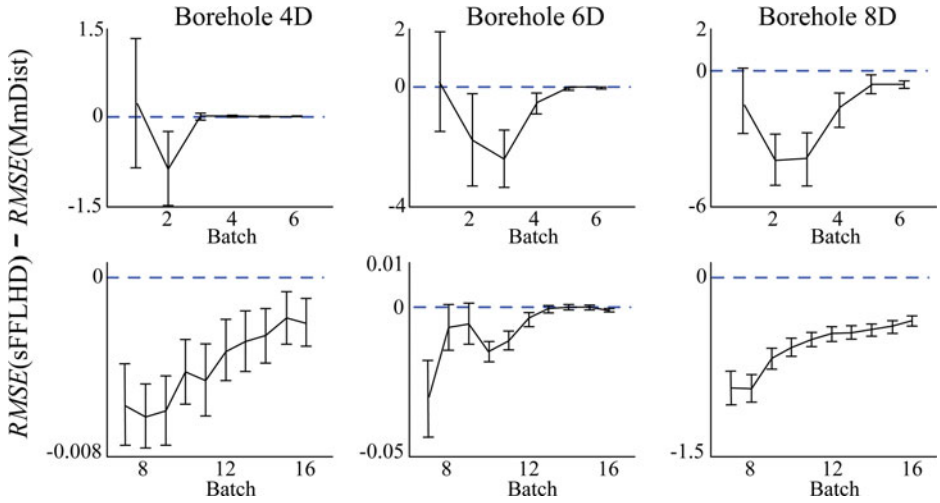


Figure 10. Borehole examples: 95% confidence intervals for RMSE(sFFLHD)-RMSE(MmDist) versus the number of batches completed, for batches 1–6 (top) and 7–16 (bottom). Dashed lines indicate differences of zero.

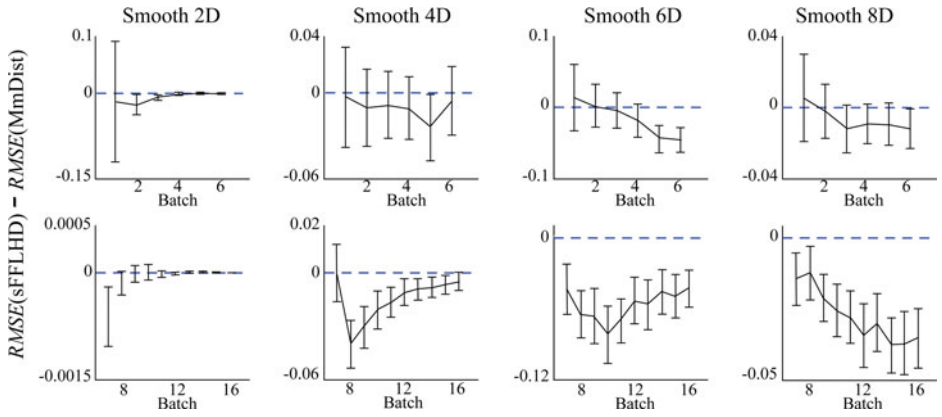


Figure 11. Smooth GP examples ($\theta = 5$): 95% confidence intervals for RMSE(sFFLHD)− RMSE(MmDist) versus the number of batches completed, for batches 1–6 (top) and 7–16 (bottom). Dashed lines indicate differences of zero.

and batches 7–16 in the second row. From the plots in Figure 10, we see that the confidence intervals are predominantly negative, and in most cases strictly negative, indicating that RMSEs from sFFLHD are significantly lower than those from MmDist. sFFLHD is most advantageous for a relatively small number of batches.

Example 2. Gaussian Process Models

For this test problem, we consider several k-dimensional Gaussian processes ($k = 2$, 4, 6, and 8). Our Gaussian model follows the form in Sacks et al. (1989) with correlation function:

$$R(\mathbf{x}, \mathbf{x}') = \exp\left(-\sum_{i=1}^{d} \theta_i (x_i - x_i')^2\right). \quad (5)$$

Different covariance parameter values, θ_i, in the Gaussian process represent different scenarios with rougher or smoother surfaces. In our surfaces, we use the same parameter value for all dimensions so $\theta_i = \theta\ \forall i$. First, we set $\theta = 5$ in each dimension. This makes the true Gaussian surface relatively smooth. RMSE differences between sFFLHD and MmDist for these GP models are shown in Figure 11. Then, we set $\theta = 15$ in each dimension, making the Gaussian surface relatively rough. Results are shown in Figure 12. Figures 11 and 12 show many instances where the confidence interval is strictly negative, indicating sFFLHD has a significantly lower RMSE than MmDist. This is particularly true for seven or more batches. All of the other confidence intervals contain zero, indicating no statistically significant difference between the designs. The overall trend shows that in very early stages, it is more difficult to distinguish any difference between the two designs. Once a few batches are observed, sFFLHD performs better than MmDist. However, as the space fills, the advantage of sFFLHD will eventually diminish because both designs begin to fit the surface well. Since the space fills faster in low dimensions, we see sFFLHD's advantage leveling off more slowly in high-dimensional, nonsmooth, examples.

6.2 Comparison With Other Designs

In addition to the MmDist design, we compare sFFLHD with several other design methods. Maximin LHD (MmLHD) is a widely used space-filling design. To implement it in a batch sequential manner, an MmLHD of the same final budget is generated in each replication and then randomly 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 budget (which is often unknown a priori), it serves as a baseline for RMSE comparison. Another design method, batch sequential LHD (bLHD) simply uses ran-

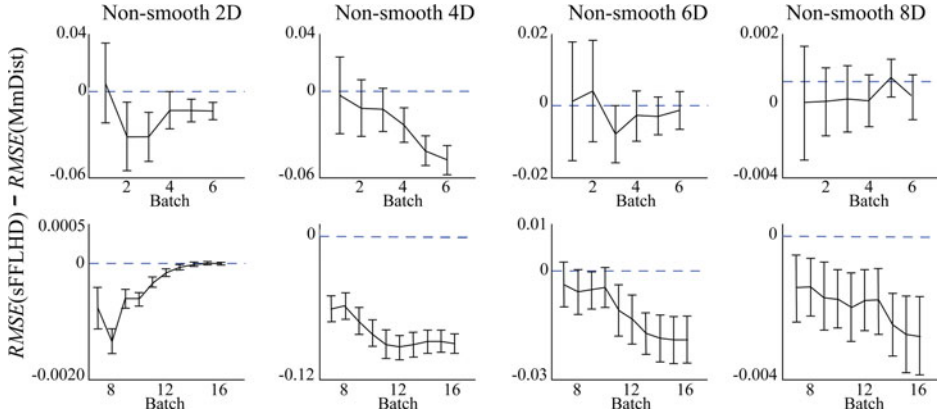


Figure 12. Nonsmooth GP examples ($\theta = 15$): 95% confidence intervals for RMSE(sFFLHD) − RMSE(MmDist) versus the number of batches completed, for batches 1–6 (top) and 7–16 (bottom). Dashed lines indicate differences of zero.

dom 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. The rows of $\mathbf{W}_{1:L^{D-1}}$ are shuffled, so that the big grid design of each batch of rsFFLHD may not be an LHD. However, at the end of Step 2, the big grid design of rsFFLHD remains an OALHD.

The online supplement provides tables that compare the RMSE of the sFFLHD method with the above design methods for Examples 1 and 2 and two additional examples that are provided in the online supplement. Any differences that are statistically significant at the 95% confidence level are shown in bold. As we found in the previous comparison, all statistically significant differences show better performance by sFFLHD. There are many instances where the differences are shown to be not statistically significant. We note that sFFLHD performs better in terms of RMSE than bMmLHD during early stages and mid-stages, and as well at the final stage. The performance of bLHD is never better than sFFLHD. As expected, rsFFLHD and sFFLHD perform equally well near stages where the big grid design of rsFFLHD is an OALHD. 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 properties at stages when sFFLHD is not an OALHD.

We also compare sFFLHD with the bin-based batch sequential design of Loeppky, Moore, and Williams (2010) on a three-dimensional GP example with $\theta = (5, 5, 5)$. The bin-based design has an initial batch of size 16 and subsequent batches of size 8, so we group batches of size 4 from an sFFLHD to match the batch sizes of the bin-based design. Because both designs have similar goals and methods, no significant difference is found between RMSEs produced from sFFLHD and the bin-based design during the 64 run experiment. However, the bin-based design does not have a clear strategy for continued experimentation beyond 64 runs, nor does it guarantee the LHD property when the bin structure (similar to the big grid design in sFFLHD) is a full factorial. Simulation shows that for each dimension, only about 89% of the 64 levels are covered when the bin-based design reaches 64 runs.

6.3 Stopping Criteria

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

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

Table 1. Borehole function: Comparison of RMSE of $\hat{\mu}$ for each design scheme

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**
rsFFLHD	223.847	34.909	**0.069**	3.502	**0.024**
bLHD	14.887	2.944	**1.660**	1.036	**0.888**

where the matrix $\mathbf{R}$ is defined using correlation function in (5) as $\mathbf{R} = \{R_{ij}\}_{(n\times n)} = R(\mathbf{x}_i., \mathbf{x}_j.)$ $\forall i, j$ and $\mathbf{r}(\mathbf{x}) = [R(\mathbf{x}, \mathbf{x}_1.), \ldots, R(\mathbf{x}, \mathbf{x}_n.)]'$. Then the root integrated MSE (RIMSE),

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

can be used as a measurement of uncertainty for prediction. Typically, the parameters θ_i, $\forall i$ are estimated and then RIMSE is approximated by computing the estimated MSE at each point on a big 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 7 to demonstrate the usage of the above stopping criteria.

6.4 Comparison of Mean Estimators

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

Example 3. Borehole Mean Estimation

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 2000 independent replications. Table 1 summarizes the result. The mean estimators of sFFLHD are superior to other designs, except when equivalent to rsFFLHD.

7. APPLICATION: OPERATIONAL AVAILABILITY SIMULATION

Our final example applies sFFLHD to a discrete-event simulation for logistics operations of interest to the U.S. Department of Defense (DoD). What follows is a brief description of the model logic. The basic scenario is that an operational unit has a fleet of vehicles. The measure of interest is the number of vehicles available at the beginning of each day, since this determines what operations can be conducted. The available proportion of the initial fleet is called the operational availability, which is abbreviated as "Ao" within DoD. All vehicles are initially in working order. Over time, vehicles are taken out of service for two reasons. The first is for periodic scheduled maintenance. A portion

of vehicles undergoing maintenance will need an extended stay in the depot because problems are identified. The second reason is that vehicles can break down prior to their scheduled maintenance. Breakdowns will immediately be placed in a queue of vehicles awaiting repair. Regularly scheduled maintenance is given precedence over breakdown repairs, based on an assumption that turnaround times will generally be quicker for the former than for the latter. Regardless of why it was in the depot, a vehicle's next maintenance cycle gets rescheduled upon departure from the depot.

Factors and their ranges of interest are:

- X_1 – Number of maintenance personnel, [2, 8];
- X_2 – Nominal ratio of initial vehicles to maintenance personnel, [5, 10];
- X_3 – Breakdown rate, [1 per 140 days, 1 per 14 days];
- X_4 – Maintenance cycle (days), [90, 120];
- X_5 – Pr(standard maintenance suffices), [0.92, 0.98];
- X_6 – Pr(standard repair required after breakdown), [0.76, 0.84];
- X_7 – Weibull scale parameter for standard repair times, [0.1, 0.5];
- X_8 – Weibull shape parameter for standard repair times, [1.5, 5].

X_1 and X_4 are integer-valued, the rest are continuous.

The standard service time is uniformly distributed between 5.5 and 6.5 hr. The standard repair time follows a Weibull distribution parameterized by X_7 and X_8. 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 choose to examine the average number of 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 eight input factors following the batch sequential method, an sFFLHD with batch size of 8 is used and GP models are fitted after each batch.

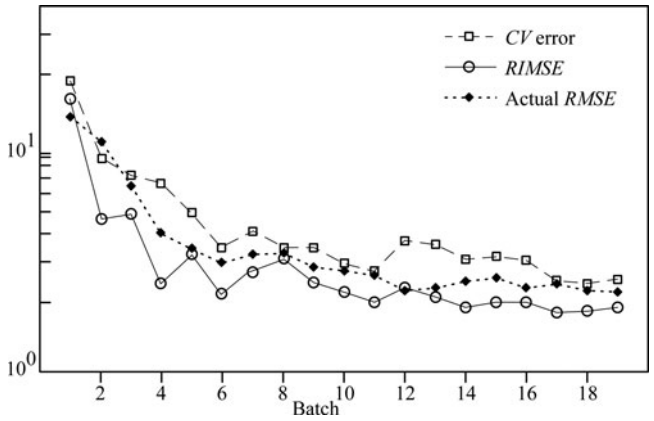


Figure 13. RIMSE, CV Error, and actual RMSE after each batch stage (in log scale).

We start with a stopping criterion based on estimated RIMSE from the GP model as shown in (7). 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 13 shows the RIMSE after each batch stage. We choose to stop after batch $b \geq 6$ if the minimum RIMSE from the five most recent batches, which we call $\text{RIMSE}_{\text{new}}$, is no more than a $p\%$ improvement over the minimum RIMSE achieved in the first $b-5$ batches, designated $\text{RIMSE}_{\text{old}}$. The criterion stops the sequential experiments if $(\text{RIMSE}_{\text{old}} - \text{RIMSE}_{\text{new}})/\text{RIMSE}_{\text{old}} < p\%$. We selected two possible scenarios ($p = 7.5$ and $p = 5$) and summarize the findings in Table 2. To assess the GP model fitting, actual model RMSEs were computed from a 10,000 test point maximin LHD (see Figure 13).

Leave-one-out cross-validation leaves each design point out in turn and refits a set of $n-1$ new surfaces. The leave-one-out cross-validation error (CV error) is a double average, first across all design points that are left in for each new surface, and then across the $n-1$ new surfaces. The CV error across batch stages are also plotted in Figure 13. Similar to the RIMSE criteria, we choose to stop after batch $b \geq 6$ if the minimum of the CV errors from the five most recent batches (CV_{new}) does not de-

Table 2. Results from RIMSE and CV error stopping criteria

Batch(b)	$\text{RIMSE}_{\text{old}}$	$\text{RIMSE}_{\text{new}}$	Relative change in RIMSE	CV_{old}	CV_{new}	Relative change in CV	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%)***				2.309

crease by more than $p\%$ of the minimum CV error from the first $b-5$ batches (CV_{old}). The criterion stops the sequential experiments if $(CV_{old} - CV_{new})/CV_{old} < p\%$. Although CV_{old} and CV_{new} both tend to decrease as more batches are used, it is likely that CV_{new} increases are due to GP model fitting errors.

Table 2 shows that the batch sequential sampling stops at batch stage 16 and 19 (in bold) 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 the RIMSE and CV error stopping criteria are able to fit GP models with small errors using a reasonable number of design points. Comparing the RIMSE with the 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 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 acquisition and logistics decisions on both readiness and life cycle cost.

8. CONCLUSION

We propose a new batch sequential design sFFLHD. At certain batch stages, sFFLHD achieves high levels of both projectivity and orthogonality by becoming fully orthogonal at the big and intermediate grid levels and becoming an LHD at the small grid level. It also achieves good sampling properties at other stages. To demonstrate its advantages, we compare it against various design methods in the context of estimating the mean and fitting a GP model to various test surfaces. When compared with many sequential design methods, sFFLHD often performs significantly better in terms of RMSEs of the GP model fit and never performs significantly worse. For estimating the mean in a region, sFFLHD produces lower variances at stages where the design is an OALHD. Empirically, we show that sFFLHD dominates the other tested designs studied in various examples provided in the article and in the supplementary materials. In addition, we examine a slight variation of sFFLHD to determine whether it is important that each batch be an LHD at the big grid level. We find this property does contribute substantially to sFFLHD's good performance even if the design does not reach the orthogonal stages. Finally, we demonstrate the use of the method and some potential stopping criteria using a simulation for vehicle availability for a fleet of vehicles.

SUPPLEMENTARY MATERIALS

Appendices A–G: Appendix A contains the proof of Proposition 1. Appendix B contains the methodology for generating nonoverlapping orthogonal arrays. Appendix C contains the methodology for generating nonoverlapping fractional factorials for use after the first golden stage. Appendix D gives the probability mass functions of the small grid design elements. Appendices E and F provide some additional empirical examples of the sFFLHD compared to other methods. Appendix G provides a table of RSME results for several examples (pdf file).

Online zip file: This file, once unzipped, contains Matlab code for implementing the sFFLHD methodology proposed in this article (two text files with .m extensions), instructions for using the Matlab code (pdf file), and a sample input file (text file).

ACKNOWLEDGMENTS

The authors thank the editor, the associate editor, and two anonymous reviewers for their constructive comments that led to great improvements in this article. The research was supported in part by the USMC-PMMI and the ONR/NPS CRUSER initiative. Department of Defense Distribution Statement: Approved for public release; distribution is unlimited.

[Received August 2013. Revised October 2015.]

REFERENCES

Aksin, Z., Armony, M., and Mehrotra, V. (2007), "The Modern Call Center: A Multi-Disciplinary Perspective on Operations Management Research," *Production and Operations Management*, 16, 665–688. [11]

Ankenman, B., Nelson, B. L., and Staum, J. (2010), "Stochastic Kriging for Simulation Metamodeling," *Operations Research*, 58, 371–382. [11]

Bernardo, M., Buck, R., Liu, L., Nazaret, W., Sacks, J., and Welch, W. (1992), "Integrated Circuit Design Optimization Using a Sequential Strategy," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 11, 361–372. [12]

Cioppa, T. M., and Lucas, T. W. (2007), "Efficient Nearly Orthogonal and Space-Filling Latin Hypercubes," *Technometrics*, 49, 45–55. [12]

Germano, M., Piomelli, U., Moin, P., and Cabot, W. H. (1991), "A Dynamic Subgrid-Scale Eddy Viscosity Model," *Physics of Fluids A: Fluid Dynamics*, 3, 1760–1765. [11]

He, X., and Qian, P. Z. G. (2011), "Nested Orthogonal Array-Based Latin Hypercube Designs," *Biometrika*, 98, 721–731. [17]

He, Y., and Ai, M. (2011), "A New Class of Latin Hypercube Designs With High-Dimensional Hidden Projective Uniformity," *Frontiers of Mathematics in China*, 6, 1085–1093. [12]

Hernandez, A. S., Lucas, T. W., and Carlyle, M. (2012), "Enabling Nearly Orthogonal Latin Hypercube Construction for any Non-Saturated Run-Variable Combination," *ACM Transactions on Modeling and Computer Simulation 20:*, 22, 1–17. [12]

Johnson, M., Moore, L., and Ylvisaker, D. (1990), "Minimax and Maximin Distance Designs," *Journal of Statistical Planning and Inference*, 26, 131–148. [12]

Joseph, V. R., and Hung, Y. (2008), "Orthogonal–Maximin Latin Hypercube Designs," *Statistica Sinica*, 18, 171–186. [12]

Journel, A. G., and Huijbregts, C. J. (1978), *Mining Geostatistics*, San Diego, CA: Academic Press. [11]

Lam, C. (2008), "Sequential Adaptive Designs in Computer Experiments for Response Surface Model Fit," Ph.D. dissertation, Ohio State University. [12]

Loeppky, J. L., Moore, L. M., and Williams, B. J. (2010), "Batch Sequential Designs for Computer Experiments," *Journal of Statistical Planning and Inference*, 140, 1452–1464. [12,19]

Matheron, G. (1963), "Principles of Geostatistics," *Economic Geology*, 58, 1246–1266. [11]

McKay, M. D., Beckman, R. J., and 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, 239–245. [11]

Morris, M. D., and Mitchell, T. J. (1995), "Exploratory Designs for Computational Experiments," *Journal of Statistical Planning and Inference*, 43, 381–402. [12]

Owen, A. (1994), "Lattice Sampling Revisited: Monte Carlo Variance of Means Over Randomized Orthogonal Arrays," *Annals of Statistics*, 22, 930–945. [16]

——— (1992), "Orthogonal Arrays for Computer Experiments, Integration and Visualization," *Statistica Sinica*, 2, 439–452. [11,17]

Qian, P. Z. G., and Wu, C. F. J. (2009), "Sliced Space-Filling Designs," *Biometrika*, 96, 945–956. [12,16]

Ranjan, P., Bingham, D., and Michailidis, G. (2008), "Sequential Experiment Design for Contour Estimation From Complex Computer Codes," *Technometrics*, 50, 527–541. [12]

Ranjan, P., and Spencer, N. (2014), "Space-Filling Latin Hypercube Designs Based on Randomization Restrictions in Factorial Experiments," *Statistics & Probability Letters*, 94, 239–247. [12]

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989), "Design and Analysis of Computer Experiments," *Statistical Science*, 4, 409–435. [18]

Tang, B. (1993), "Orthogonal Array-Based Latin Hypercubes," *Journal of the American Statistical Association*, 88, 1392–1397. [11,17]

Worley, B. (1987), "Deterministic Uncertainty Analysis," ORNL-6428, Oak Ridge National Laboratory, Oak Ridge, Tennessee. [17]