Statistica Sinica **24** (2014), 1239-1256
doi:http://dx.doi.org/10.5705/ss.2012.347

GENERAL SLICED LATIN HYPERCUBE DESIGNS

Huizhi Xie, Shifeng Xiong, Peter Z. G. Qian and C. F. Jeff Wu

Netflix, Chinese Academy of Sciences,
University of Wisconsin-Madison and Georgia Institute of Technology

Abstract: We propose a method for constructing general sliced Latin hypercube designs, intended for computer experiments. A general sliced Latin hypercube design has multiple layers, at each of which there are multiple Latin hypercube designs that can be sliced into smaller Latin hypercube designs at the next layer. The proposed method is easy to implement, capable of accommodating any number of factors and flexible in run size. Such designs include ordinary Latin hypercube design and sliced Latin hypercube design as special cases. A special case of general sliced Latin hypercube design with two layers, called doubly sliced Latin hypercube design, is studied in detail, including its sampling properties. The more flexible structure of doubly sliced Latin hypercube design allows more flexible batch size for both collective evaluation of different computer models and batch sequential evaluation of a single computer model. Numerical examples are provided to show its advantages.

Key words and phrases: Computer experiment, design of experiments, space-filling design, ensembles of computer models.

1. Introduction

Experiments with deterministic simulation codes have become ubiquitous in science, engineering, and services for studying complex phenomena. Here, running a deterministic simulation with the same inputs yields identical outputs (Santner, Williams, and Notz (2003); Fang, Li, and Sudjianto (2005)). A goal in many computer experiments is to estimate the expected output of a computer model given a distribution of inputs. To address this, McKay, Conover, and Beckman (1979) introduced Latin hypercube designs (LHD), referred to as *ordinary Latin hypercube designs* hereinafter. Qian (2012) proposed sliced Latin hypercube designs for collective evaluation of computer models. Such a design is a special LHD that can be decomposed into smaller slices, each of which is an LHD. This slicing idea is related to central composite designs for quantitative and qualitative factors (Wu and Ding (1998)).

Sometimes one is interested in collective evaluation of related computer models. For example, we may have nine computer models A_1, A_2, A_3, B_1, B_2, B_3 and

C_1, C_2, C_3, where A_is are variants of a finite element model A implemented with different mesh densities, B_is inherit a finite element model B with different mesh densities, and C_is come from a finite element model C with different mesh densities. Moreover, A, B, and C are related to each other. We are interested in estimating the expected output of each of the nine computer models, a linear combination of the expected outputs of the variants for each of A, B, and C, as well as a linear combination of the expected outputs of the nine computer models. An example of computer models solved using different mesh densities is in Tuo, Wu, and Yu (2013). There the computer model simulating a casting process is solved using different mesh densities, considered as a tuning parameter. As another example, we may have three computer models A, B, and C and we want to run each of the models in multiple batches to estimate the expected outputs of each of them, as well as a linear combination of the expected outputs of the three models. Running the computer models in batches may enable us to stop earlier once we have achieved a desired accuracy for the estimator. To achieve more flexible slicing, we propose a new class of designs called general sliced Latin hypercube designs (GSLHD). These designs are useful for collective ensemble and batch evaluation of computer models.

The remainder of the article is organized as follows. Section 2 discusses the construction of a general sliced Latin hypercube design. Some sampling properties of GSLHD with double slicing are derived in Section 3. Section 4 provides numerical illustration of some advantages of the proposed design. Summary and future research directions are presented in Section 5. The supplementary file contains proofs.

2. Construction of General Sliced Latin Hypercube Design

We first review the construction of a sliced Latin hypercube design (SLHD), and then present a recursive strategy for constructing a GSLHD. The stepping stone of the construction of an SLHD in Qian (2012) is the generation of sliced permutation matrices. For positive integers n, m, t with $n = mt$, a permutation matrix PM(m, t) on $\boldsymbol{Z}_n = \{1, \ldots, n\}$ is an $m \times t$ matrix in which each element of $\boldsymbol{Z}_n$ appears exactly once. If $\boldsymbol{A}$ is a PM(m, t) and each column of $\lceil \boldsymbol{A}/t \rceil$ forms a permutation on $\boldsymbol{Z}_m$, we call $\boldsymbol{A}$ an $m \times t$ $s$liced permutation matrix, denoted by SPM(m, t). It can be used to construct an SLHD of t slices, each of which has m runs.

For positive integers m, s_1, s_2, and d, a doubly sliced Latin hypercube design (DSLHD) $\boldsymbol{D}$ is an LHD with ms_2s_1 levels in d dimensions. Such a $\boldsymbol{D}$ can be partitioned into $\boldsymbol{D}_{ij}$'s as

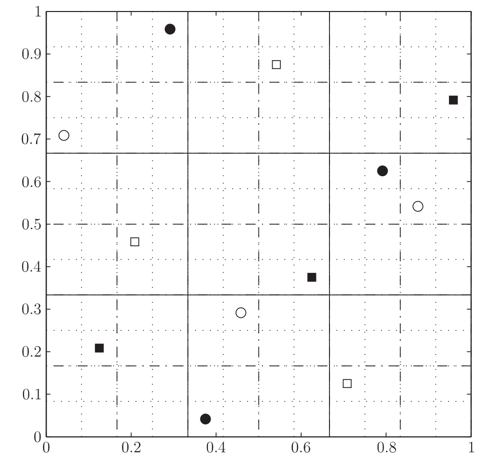

Figure 1. An example of DSLHD. All markers in the figure constitute a DSLHD(3,2,2,2). The design can be decomposed into $\boldsymbol{D}_{11}$, $\boldsymbol{D}_{12}$, $\boldsymbol{D}_{21}$ and $\boldsymbol{D}_{22}$ as in (2.1). The three stuffed circles are from $\boldsymbol{D}_{11}$. The three stuffed squares correspond to $\boldsymbol{D}_{12}$. The three unstuffed circles belong to $\boldsymbol{D}_{21}$. The three unstuffed squares come from $\boldsymbol{D}_{22}$.

$$\begin{matrix} \boldsymbol{D}_{11} & \boldsymbol{D}_{12} & \cdots & \boldsymbol{D}_{1s_1} \\ \boldsymbol{D}_{21} & \boldsymbol{D}_{22} & \cdots & \boldsymbol{D}_{2s_1} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{D}_{s_21} & \boldsymbol{D}_{s_22} & \cdots & \boldsymbol{D}_{s_2s_1} \end{matrix}, \tag{2.1}$$

where $\boldsymbol{D}_{ij}$ is a d-dimensional LHD with m levels for $i = 1, \ldots, s_2,\ j = 1, \ldots, s_1$, $\boldsymbol{D}_i = \cup_{j=1}^{s_1} \boldsymbol{D}_{ij}$ is a d-dimensional LHD with ms_1 levels for $i = 1, \ldots, s_2$. Compared with an SLHD, each slice $\boldsymbol{D}_i$ in a DSLHD can be further partitioned into s_1 smaller LHDs. For $s_1 = 1$, a DSLHD becomes an SLHD. Throughout, we use DSLHD(m, s_2, s_1, d) to denote a DSLHD associated with parameters m, s_2, s_1, and d. An example of a DSLHD(3,2,2,2) is shown in Figure 1.

The sliced structure can be extended. Given a positive integer r, for positive integers n, m, $s_1, \ldots, s_r$ with $n = m\prod_{i=1}^{r} s_i$, an r-layer general sliced Latin hypercube design $\boldsymbol{D}$ has the following form. In the first layer, $\boldsymbol{D}$ consists of $\prod_{k=1}^{r} s_k$ LHDs, each of which has m runs; in the second layer, there are $\prod_{k=2}^{r} s_k$ LHDs, each of which is of ms_1 levels and consists of s_1 LHDs from the first layer; in the kth layer for $k = 3, \ldots, r$, there are $\prod_{j=k}^{r} s_j$ LHDs, each of which is of $m\prod_{j=1}^{k-1} s_j$ levels and consists of s_k LHDs from the $(k-1)$th layer. The s_r LHDs in the rth layer constitute the whole design, which is also an LHD. Such a design is denoted by GSLHD($s_1, \ldots, s_r$; m; d), where $s_1, \ldots, s_r$ represent the layer structure, m is the number of runs in each LHD in the first layer and d is the number of factors. A GSLHD($s_1, s_2, \ldots, s_r$; m; d) reduces to an SLHD and

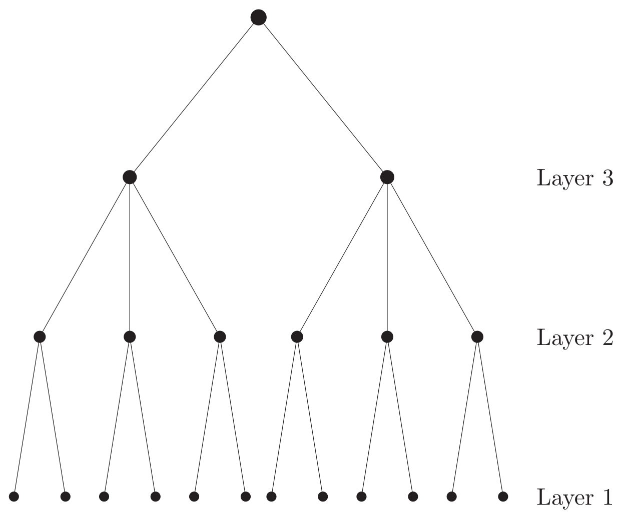

Figure 2. Tree diagram for a three-layer SLHD with $s_1 = 2$, $s_2 = 3$ and $s_3 = 2$, where each node denotes an LHD. There are 12 ($= s_1s_2s_3$) small LHDs in layer 1. In layer 2, each LHD can be partitioned into two ($= s_1$) LHDs in layer 1. In layer 3, each LHD can be partitioned into three ($= s_2$) LHDs in layer 2. The two ($= s_3$) LHDs in layer 3 constitute the whole design.

a DSLHD for $r = 1$ and $r = 2$, respectively. An ordinary LHD without any sliced structure can be viewed as an r-layer general sliced Latin hypercube design with $r = 0$. An example of the layer structure of a general sliced Latin hypercube design of three layers with $s_1 = 2$, $s_2 = 3$ and $s_3 = 2$ is shown in Figure 2.

We consider design construction for d quantitative factors each taking value in $(0, 1]$. For a real number a, let $\lceil a \rceil$ denote the smallest integer greater than or equal to a and $\lfloor a \rfloor$ denote the largest integer less than or equal to a. Similarly define $\lceil \boldsymbol{A} \rceil$ and $\lfloor \boldsymbol{A} \rfloor$ for a real matrix $\boldsymbol{A}$. For a real matrix $\boldsymbol{A}$, let $\boldsymbol{A}(:, j)$ be its jth column, $\boldsymbol{A}(i, :)$ its ith row and $\boldsymbol{A}(i, j)$ its $(i,\ j)$th element. For an integer $b \geq 1$, let $\boldsymbol{Z}_b$ denote the set $\{1, \ldots, b\}$. Drawing a uniform permutation on a set of b integers means taking a permutation on the set with all $b!$ possible permutations equally probable.

Qian (2012) used SPM for constructing an SLHD. We use sliced permutation vectors instead of permutation matrices. An SPM(m, t) can be written as a one-layer SPV(t; m), a concatenation of the columns of an SPM(m, t). A one-layer SPV(t; m) is a permutation on $\boldsymbol{Z}_n$, let $\{p_1, \ldots, p_n\}$ be its elements. For $i = 1, \ldots, t$, $\left\{\lceil p_{(i-1)m+1}/t \rceil, \ldots, \lceil p_{(i-1)m+m}/t \rceil\right\}$ is a permutation on $\boldsymbol{Z}_m$. These properties follow immediately from those of an SPM(m, t). More generally for $r > 1$, an r-layer sliced permutation vector SPV($s_1, \ldots, s_r$; m) with entries

$\{p_1,\ldots,p_n\}$ is a permutation on $\boldsymbol{Z}_n$, where for $i=1,\ldots,s_r$, $\big\{\lceil p_{(i-1)w+1}/s_r\rceil,\ldots,$ $\lceil p_{(i-1)w+w}/s_r\rceil\big\}$ is an $(r-1)$-layer SPV$(s_1,\ldots,s_{r-1};\ m)$. Here $w=m\prod_{k=1}^{r-1}s_k$. Consider the three-layer general sliced Latin hypercube design in Figure 2 as an example. To construct a GSLHD$(2,3,2;m;\ d)$, we first generate SPV$(2,3,2;\ m)$'s. An SPV$(2,3,2;\ m)$ can be partitioned into segments $(\mathbf{b}_1',\mathbf{b}_2')$ corresponding to two LHDs in the third layer, where $\lceil\mathbf{b}_1/2\rceil$ and $\lceil\mathbf{b}_2/2\rceil$ have to be two SPV$(3,2;$ $m)$'s. Therefore, to generate an SPV$(2,3,2;\ m)$, we can first generate two SPV$(2,3;\ m)$'s. Similarly, to generate an SPV$(2,3;\ m)$, we can generate three SPV$(2;\ m)$'s, which can be generated using the algorithm in Qian (2012).

We present a recursive strategy for generating SPV$(s_1,\ldots,s_r;\ m)$.

```
Step 1: Construct a $w \times s_r$ matrix $\boldsymbol{H} = (h_{ij})$ whose $i$th row is $\big((i-1)s_r+1,\ldots,(i-1)s_r+s_r\big)$. Here $w = m\prod_{k=1}^{r-1} s_k$.
Step 2: Construct a $w \times s_r$ matrix $\boldsymbol{C} = (c_{ij})$ whose $i$th row is a uniform permutation on $\big(h_{i1},\ldots,h_{is_r}\big)$. Permutations are carried out independently from one row to another.
Step 3: For $i = 1,\ldots,s_r$,
            If $r = 1$,
              Generate a uniform permutation $(p_1,\ldots,p_w)'$ on $\boldsymbol{Z}_w$.
            Else
              Suppose an SPV$(s_1,\ldots,s_{r-1};\ m)$=$(p_1,\ldots,p_w)'$ has been generated.*
              Construct a permutation $\mathbf{b}_i = (b_{1i},\ldots,b_{wi})'$ on $(c_{1i},\ldots,c_{wi})'$
              such that
                          $b_{ji} = c_{p_j i}$ for $j = 1,\ldots,w$.
            End If
        End $i$ loop.**
Step 4: An SPV$(s_1,\ldots,s_r;\ m)$ is given by $(\mathbf{b}_1',\ldots,\mathbf{b}_{s_r}')'$.***
```

Note * This is where the recursive strategy comes in. To generate an SPV $(s_1,\ldots,\ s_{r-1};\ m)$, we need to independently generate s_{r-1} SPV$(s_1,\ldots,$ $s_{r-2};\ m)$'s. This recursion proceeds until $r=1$, where SPM$(s_1;\ m)$ can be generated using the algorithm in Qian (2012).

** In this step, the implementations in all iterations of the i loop are carried out independently.

*** When $r=2$, the SPV$(s_1,s_2;\ m)$ generated can be used for constructing a DSLHD.

As pointed out by a referee, one can also use the orthogonal array-based method in Tang (1993) to construct general sliced Latin hypercube designs. After generating s independent permutations of $\boldsymbol{Z}_m$, we replace all positions with entry i by a permutation of $\{(i-1)s+1,\ldots,(i-1)s+s\}$ for each $i=1,\ldots,m$. The new permutation, $\boldsymbol{Z}_{ms}$, is an SPV$(s;m)$. An SPV$(s_1,\ldots,s_r;\ m)$ can be constructed by a recursive strategy. Generating s_r independent SPV$(s_1,\ldots,s_{r-1};\ m)$'s, then replacing all positions with entry i by a permutation of $\{(i-1)s_r+1,\ldots,(i-1)s_r+s_r\}$ for each $i=1,\ldots,s_1\cdots s_{r-1}m$ produces an SPV$(s_1,\ldots,s_r;\ m)$.

Example 1. We generate a two-layer SPV$(2,2;\ 3)$. Suppose the matrix $\boldsymbol{C}$ (in transpose) in Step 2 is

$$\boldsymbol{C}' = \begin{pmatrix} 1 & 4 & 6 & 7 & 9 & 12 \\ 2 & 3 & 5 & 8 & 10 & 11 \end{pmatrix}.$$

For Step 3, since $s_2=2$, we need to generate two independent SPV$(2;\ 3)$'s. Using Qian (2012)'s algorithm, suppose we have generated $(1,3,6,4,2,5)'$ and $(1,5,4,6,2,3)'$. Note both SPV$(2;\ 3)$'s are concatenation of vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ such that $\lceil \boldsymbol{v}_k/2\rceil$ is a permutation vector of an ordinary LHD of three levels. Here $k=1,2$. Given the generated SPV$(2;\ 3)$s, $\mathbf{b}_1$ and $\mathbf{b}_2$ in Step 3 are $(1,6,12,7,4,9)'$ and $(2,10,8,11,3,5)'$, respectively. The desired two-layer SPV$(2,2;\ 3)$ is $(1,6,12,7,4,9,2,10,8,11,3,5)'$.

Example 2. We generate an SPV$(2,2,2;\ 3)$. Suppose the matrix $\boldsymbol{C}$ (in transpose) in Step 2 is

$$\boldsymbol{C}' = \begin{pmatrix} 1 & 4 & 6 & 7 & 10 & 11 & 13 & 15 & 18 & 20 & 21 & 24 \\ 2 & 3 & 5 & 8 & 9 & 12 & 14 & 16 & 17 & 19 & 22 & 23 \end{pmatrix}.$$

Since $s_3=2$, we need to generate two independent SPV$(2,2;\ 3)$. As in Example 1, suppose we have generated $(7,2,10,12,4,5,1,9,8,11,6,3)'$ and $(6,1,12,9,4,7,8,2,10,11,3,5)'$. $\mathbf{b}_1$ and $\mathbf{b}_2$ in Step 3 are thus $(13,4,20,24,7,10,1,18,15,21,11,6)'$ and $(12,2,23,17,8,14,16,3,19,22,5,9)'$, respectively. Hence the desired SPV $(2,2,\ 2;3)$ is $(13,4,20,24,7,10,1,18,15,21,11,6,12,2,23,17,8,14,16,3,19,22,5,9)'$. The generated $SPV(2,2,2;\ 3)$ can be partitioned into eight vectors: $\mathbf{v}_{111}=(13,4,20)'$, $\mathbf{v}_{112}=(24,7,10)'$, $\mathbf{v}_{121}=(1,18,15)'$, $\mathbf{v}_{122}=(21,11,6)'$, $\mathbf{v}_{211}=(12,2,23)'$, $\mathbf{v}_{212}=(17,8,14)'$, $\mathbf{v}_{221}=(16,3,19)'$ and $\mathbf{v}_{222}=(22,5,9)'$. At the first layer, $\lceil \mathbf{v}_{ijk}/8\rceil$ is a permutation on $\boldsymbol{Z}_3$, $i,j,k=1,2$. At the second layer, let $\mathbf{v}_{11}=(\mathbf{v}'_{111},\mathbf{v}'_{112})'=(13,4,20,24,7,10)'$, $\mathbf{v}_{12}=(\mathbf{v}'_{121},\mathbf{v}'_{122})'=(1,18,15,21,11,6)'$, $\mathbf{v}_{21}=(\mathbf{v}'_{211},\mathbf{v}'_{212})'=(12,2,23,17,8,14)'$ and $\mathbf{v}_{22}=(\mathbf{v}'_{221},\mathbf{v}'_{222})'=(16,3,19,22,5,9)'$. $\lceil \mathbf{v}_{ij}/4\rceil$ is a permutation on $\boldsymbol{Z}_6$, $i,j=1,2$. At the third layer, let $\mathbf{v}_1=(\mathbf{v}'_{11},\mathbf{v}'_{12})'=(13,4,20,24,7,10,1,18,15,21,11,6)$ and $\mathbf{v}_2=(\mathbf{v}'_{21},\mathbf{v}'_{22})'=(12,2,23,17,8,14,16,3,19,22,5,9)$. $\lceil \mathbf{v}_i/2\rceil$ is a permutation on $\boldsymbol{Z}_{12}$.

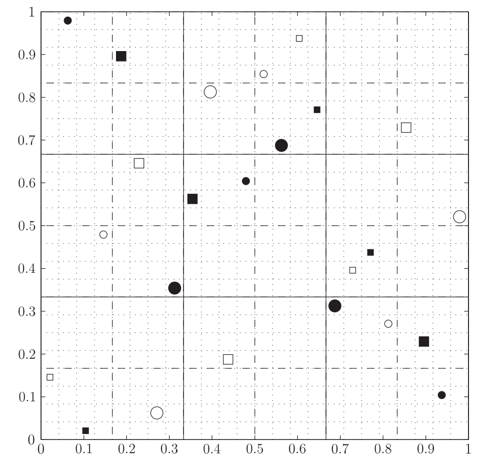

Figure 3. An example of three-layer GSLHD. All markers in the figure constitute a GSLHD$(2,2,2;3;\ 2)$. In the first layer, the GSLHD$(2,2,2;3;\ 2)$ is partitioned into eight small LHDs, denoted by small stuffed circles, large stuffed circles, small stuffed squares, large stuffed squares, small unstuffed circles, large unstuffed circles, small unstuffed squares, and large unstuffed squares, respectively. Each of the small LHDs has three runs. In the second layer, there are four LHDs, denoted by stuffed circles, stuffed squares, unstuffed circles, and unstuffed squares, respectively. Each LHD in this layer can be partitioned into two LHDs in the first layer. In the third layer, there are two LHDs, denoted by stuffed markers and unstuffed markers, respectively. Each LHD in this layer can be partitioned into two LHDs in the second layer.

Based on r-layer SPV$(s_1,\ldots,s_r;\ m)$'s, an r-layer GSLHD$(s_1,\ldots,s_r;m;\ d)$ can be generated as follows.

Step 1: Construct an $n \times d$ matrix $\boldsymbol{A} = (a_{ij})$, whose columns are d independent SPV$(s_1,\ldots,s_r;\ m)$s. Here $n = m\prod_{i=1}^{r} s_i$.

Step 2: Construct an $n \times d$ matrix $\boldsymbol{D}$, the (i,j)th entry of which is

$$d_{ij} = \frac{a_{ij} - u_{ij}}{n}, \quad \text{for } i = 1,\ldots,n,\ j = 1,\ldots,d,$$

where u_{ij} are i.i.d. $U[0,1)$ random variables, and a_{ij} and u_{ij} are mutually independent. $\boldsymbol{D}$ is an r-layer GSLHD$(s_1,\ldots,s_r;\ m;\ d)$.

An example of a three-layer GSLHD is shown in Figure 3.

3. Sampling Properties

We derive sampling properties of the DSLHD. Unlike an SLHD, a DSLHD can be sliced twice: a bigger slice at the first layer can be further sliced into smaller LHD's at the second layer. This enables us to allocate LHDs of smaller size to different computer models for collective evaluation. Moreover, when we combine these designs, they form a larger LHD. Hence DSLHD enables a spacing property at a finer scale than SLHD. This shows the advantage of DSLHD over SLHD for collective evaluation of computer models. This will be seen more clearly in the numerical experiments. A list of schemes for collective evaluation of different computer models is given in Definition 1. Specific examples will be given to illustrate the greater flexibility of DSLHD. First, we define the collective evaluation problem as follows.

Suppose there are s_2 computer models and each of them has s_1 variants. For example, we can have s_2 different finite element models and each of them is solved using s_1 different mesh densities. Denote the computer models as f_{ij}, $i = 1, \ldots, s_2$, $j = 1, \ldots, s_1$, where f_{ij} is the jth variant of the ith computer model. Assume each f_{ij} has factors $\boldsymbol{x} = (x_1, \ldots, x_d)$ with the uniform distribution F on $(0, 1]^d$. For $c_1 = 1, \ldots, s_1$, $c_2 = 1, \ldots, s_2$, take $\mu_{c_2c_1} = \mathrm{E}[f_{c_2c_1}(\boldsymbol{x})]$. For c_{11}, $c_{12} = 1, \ldots, s_1$, $c_{21}, c_{22} = 1, \ldots, s_2$, take $\mathrm{cov}_{c_{21}c_{11}c_{22}c_{12}} = \mathrm{cov}[f_{c_{21}c_{11}}(\boldsymbol{x}), f_{c_{22}c_{12}}(\boldsymbol{x})]$, which becomes $\sigma^2_{c_{21}c_{11}} = \mathrm{var}[f_{c_{21}c_{11}}(\boldsymbol{x})]$ if $c_{21} = c_{22}$ and $c_{11} = c_{12}$. The goal here is to run each f_{ij} at m selected input values for the purpose of estimating μ_{ij}. For $0 \le \lambda_{ij} \le 1$, $i = 1, \ldots, s_2$, $j = 1, \ldots, s_1$, the following linear combinations are of interest:

$$\eta = \sum_{i=1}^{s_2} \sum_{j=1}^{s_1} \lambda_{ij}\mu_{ij},$$

$$\eta_i = \sum_{j=1}^{s_1} \lambda_{ij}\mu_{ij}, \quad i = 1, \ldots, s_2. \tag{3.1}$$

Definition 1. Suppose m, s, s_1, and s_2 are positive integers with $n = ms = ms_1s_2$.

(i) IID is a scheme that takes an independent and identically distributed sample of m runs for each f_{ij}, with the s samples generated independently.

(ii) LH is a scheme that obtains s independent ordinary Latin hypercube designs of m runs, each of which is associated with one f_{ij}.

(iii) SLH-ORG (original SLH) is a scheme that produces an $n \times d$ SLHD with s slices by using the method in Qian (2012), where each slice is a smaller LHD with m levels and is assigned to one f_{ij}.

(iv) SLH-IND(independent SLH) is a scheme that independently produces s_2 $ms_1 \times d$ SLHD's, each of which has s_1 slices using the method in Qian (2012). For $i = 1, \ldots, s_2$, $j = 1, \ldots, s_1$, jth slice in the ith SLHD is assigned to f_{ij}.

(v) SLH-SPL(split slices of SLH) is a scheme that generates an $n \times d$ SLHD with s_2 slices using the method in Qian (2012), where each slice is a smaller LHD with ms_1 levels. For $i = 1, \dots, s_2$, the ith slice is randomly split into s_1 subsets of size m and each of the subsets is assigned to one variant of the ith computer model.

(vi) DSLH is a scheme that produces a DSLHD(m,s_2,s_1,d), with, for $i = 1, \dots, s_2$, $j = 1, \dots, s_1$, D_{ij} assigned to f_{ij}.

Expectation, variance and covariance under these schemes (in the same order) are denoted by the subscripts IID, LH, SLH-ORG, SLH-IND, SLH-SPL and DSLH, respectively. For any of these schemes, let $\boldsymbol{D}_{ij}$ denote the design set for f_{ij}, $i = 1, \dots, s_2$, $j = 1, \dots, s_1$. Denote by $\boldsymbol{d}_{ij}^{(c)}$ the cth row of $\boldsymbol{D}_{ij}$ and $d_{ij}^{(ck)}$ the (c, k)th entry of $\boldsymbol{D}_{ij}$. For $i = 1, \dots, s_2$, $j = 1, \dots, s_1$, μ_{ij} is estimated by

$$\hat{\mu}_{ij} = m^{-1} \sum_{c=1}^{m} f_{ij}(\boldsymbol{d}_{ij}^{(c)}), \tag{3.2}$$

η_i by

$$\hat{\eta}_i = \sum_{j=1}^{s_1} \lambda_{ij} \hat{\mu}_{ij}, \tag{3.3}$$

and η by

$$\hat{\eta} = \sum_{i=1}^{s_2} \sum_{j=1}^{s_1} \lambda_{ij} \hat{\mu}_{ij}. \tag{3.4}$$

For later development, we describe the ANOVA decomposition of integrable functions on $[0, 1]^d$ (Owen (1992); Loh (1996)). With F the uniform measure on $(0, 1]^d$, let $dF = \prod_{k=1}^{d} dF_k$ and $dF_{-k} = \prod_{l \neq k} dF_l$. If f:$R^d \to R$ is a measurable function of $\boldsymbol{x} = (x_1, \dots, x_d)$ and $\mathrm{E}[f(\boldsymbol{x})]^2$ is well defined and finite, then f can be decomposed as

$$f(\boldsymbol{x}) = \mu + \sum_{k=1}^{d} f_{-k}(x_k) + r(\boldsymbol{x}), \tag{3.5}$$

where $\mu = \int f(\boldsymbol{x}) dF$ is the grand mean and the functional main effect of x_k is

$$f_{-k}(x_k) = \int [f(\boldsymbol{x}) - \mu] dF_{-k}. \tag{3.6}$$

For $k = 1, \dots, d$, $\int f_{-k} dF_k = 0$ and

$$\int r(\boldsymbol{x}) dF_{-k} = 0. \tag{3.7}$$

For an SPM(s_1, s_2; m) associated with a DSLHD(m,s_2,s_1,1), the vector can be partitioned into $s_1 s_2$ segments, each of which has m elements and is assigned to one of the $\boldsymbol{D}_{ij}$s in (2.1) in the order $\boldsymbol{D}_{11}, \dots, \boldsymbol{D}_{1s_1}, \dots, \boldsymbol{D}_{s_2 s_1}$. For ease of notation, define an $m \times s_2 \times s_1$ doubly sliced permutation matrix (DSPM) such that DSPM(:,i,j) is the vector assigned to $\boldsymbol{D}_{ij}$. We present some results on the joint probability mass functions of the elements of a DSPM, beginning with results on SLHD, Lemmas 1 and 2 in Qian (2012).

Lemma 1. *Let $\boldsymbol{H}$ be a SPM(m,t) on $\boldsymbol{Z}_n$, where $n=mt$, and h_{ij} is the (i,j)th entry of $\boldsymbol{H}$. For u, $v \in \boldsymbol{Z}_n$, we have the following*

(i) *For $i=1,\ldots,m$, $j=1,\ldots,t$, the probability mass function for h_{ij} is*

$$Pr(h_{ij}=u)=\frac{1}{n}.$$

(ii) *For i_1, $i_2=1,\ldots,m$, $i_1\neq i_2$ and $j=1,\ldots,t$, the joint probability mass function for h_{i_1j} and h_{i_2j} is*

$$Pr(h_{i_1j}=u,h_{i_2j}=v)=\begin{cases}[n(n-t)]^{-1}, & \lceil\frac{u}{t}\rceil\neq\lceil\frac{v}{t}\rceil,\\ 0, & otherwise.\end{cases} \tag{3.8}$$

(iii) *For i_1, $i_2=1,\ldots,m$, j_1, $j_2=1,\ldots,t$, $j_1\neq j_2$, the joint probability mass function for $h_{i_1j_1}$ and $h_{i_2j_2}$ is*

$$Pr(h_{i_1j_1}=u,h_{i_2j_2}=v)=\begin{cases}n^{-2}, & \lceil\frac{u}{t}\rceil\neq\lceil\frac{v}{t}\rceil,\\ [n(n-m)]^{-1}, & \lceil\frac{u}{t}\rceil=\lceil\frac{v}{t}\rceil \text{ and } u\neq v,\\ 0, & otherwise.\end{cases} \tag{3.9}$$

Lemma 2. *Let $\boldsymbol{D}$ be an SLHD with t slices, $\boldsymbol{D}_1,\ldots,\boldsymbol{D}_t$. Then $\boldsymbol{D}_c$ is statistically equivalent to an $m\times d$ ordinary Latin hypercube design for each $c=1,\ldots,t$.*

Now we present results on DSPM.

Lemma 3. *Let $\boldsymbol{H}$ be a DSPM and let h_{ijk} be its (i,j,k)th entry. For u, v, $w\in\boldsymbol{Z}_n$, let*

$$B_1=\Big\{(u,v)|\lceil\frac{u}{s_2s_1}\rceil\neq\lceil\frac{v}{s_2s_1}\rceil\Big\}; \tag{3.10}$$

$$B_2=\Big\{(u,v)|\lceil\frac{u}{s_2s_1}\rceil=\lceil\frac{v}{s_2s_1}\rceil,\lceil\frac{(u-\lfloor u/s_2s_1\rfloor)}{s_2}\rceil\neq\lceil\frac{(v-\lfloor v/s_2s_1\rfloor)}{s_2}\rceil\Big\}; \tag{3.11}$$

$$B_3=\Big\{(u,v)|\lceil\frac{u}{s_2s_1}\rceil=\lceil\frac{v}{s_2s_1}\rceil,\lceil\frac{(u-\lfloor u/s_2s_1\rfloor)}{s_2}\rceil=\lceil\frac{(v-\lfloor v/s_2s_1\rfloor)}{s_2}\rceil\Big\}. \tag{3.12}$$

(i) *For $i=1,\ldots,m$, $j=1,\ldots,s_2$, $k=1,\ldots,s_1$, the probability mass function for h_{ijk} is*

$$Pr(h_{ijk}=u)=\frac{1}{n}. \tag{3.13}$$

(ii) *For i_1, $i_2 = 1, \ldots, m$, $i_1 \neq i_2$, and $j = 1, \ldots, s_2$, $k = 1, \ldots, s_1$, the joint probability mass function for h_{i_1jk} and h_{i_2jk} is*

$$Pr(h_{i_1jk} = u, h_{i_2jk} = v) = \begin{cases} [n(n - s_2s_1)]^{-1}, & (u, v) \in B_1, \\ 0, & otherwise. \end{cases} \tag{3.14}$$

(iii) *For i_1, $i_2 = 1, \ldots, m$, $j = 1, \ldots, s_2$, k_1, $k_2 = 1, \ldots, s_1$, $k_1 \neq k_2$, the joint probability mass function for $h_{i_1jk_1}$ and $h_{i_2jk_2}$ is*

$$Pr(h_{i_1jk_1} = u, h_{i_2jk_2} = v) = \begin{cases} [n(n - s_2m)]^{-1}, & (u, v) \in B_2, \\ n^{-2}, & (u, v) \in B_1, \\ 0, & otherwise. \end{cases} \tag{3.15}$$

(iv) *For i_1, $i_2 = 1, \ldots, m$, j_1, $j_2 = 1, \ldots, s_2$, $j_1 \neq j_2$, k_1, $k_2 = 1, \ldots, s_1$, the joint probability mass function for $h_{i_1j_1k_1}$ and $h_{i_2j_2k_2}$ is*

$$Pr(h_{i_1j_1k_1} = u, h_{i_2j_2k_2} = v) = \begin{cases} n^{-2}, & (u, v) \in B_1 \text{ or } B_2, \\ [n(n - ms_1)]^{-1}, & (u, v) \in B_3, \\ 0, & otherwise. \end{cases} \tag{3.16}$$

When $s_1 = 1$, (ii) and (iv) reduce to (ii) and (iii) of Lemma 1 in Qian (2012), respectively, so the present result can be viewed as an extension of that result.

Let $\boldsymbol{D}$ be an $n \times d$ DSLHD with parameters m, s_2, s_1, and d, and $\boldsymbol{D}_{rc}$ be as in (2.1), where $r = 1, \ldots, s_2$, and $c = 1, \ldots, s_1$. The sampling distribution of $\boldsymbol{D}_{rc}$ and $\boldsymbol{D}_r = \bigcup_{c=1}^{s_1} \boldsymbol{D}_{rc}$ is given next.

Lemma 4. *For positive integers m, s_2, and s_1 with $n = ms_2s_1$, $\boldsymbol{D}_{rc}$ at* (2.1) *is statistically equivalent to an $m \times d$ ordinary Latin hypercube design; $\boldsymbol{D}_r$ is statistically equivalent to an SLHD* (*constructed using the method in Qian (2012)*) *of s_1 slices, each of which is an ordinary Latin hypercube design with m levels.*

Lemma 5. *For a DSLHD($m, s_2, s_1, 1$) $\boldsymbol{D}$, the covariance between any point in $\boldsymbol{D}_{i_1j_1}$ and any point in $\boldsymbol{D}_{i_2j_2}$, $i_1 \neq i_2$, is non-positive.*

We have results on $\hat{\mu}_{ij}$ in (3.2), $\hat{\eta}_i$ in (3.3), and $\hat{\eta}$ in (3.4) under doubly sliced Latin hypercube sampling.

Theorem 1. *Suppose that, for $i = 1, \ldots, s_2$, $j = 1, \ldots, s_1$, $f_{ij}(\boldsymbol{x})$ is monotonic in each argument x_k of $\boldsymbol{x} = (x_1, \ldots, x_d)$, and any pair of functions $f_{i_1j_1}$ and $f_{i_2j_2}$ is jointly increasing or decreasing in each argument x_k of $\boldsymbol{x}$. For the six schemes of Definition 1, we have the following.*

(i) *For $i = 1, \ldots, s_2$, $j = 1, \ldots, s_1$, and $\hat{\mu}_{ij}$,*

$$\begin{aligned} var_{\text{DSLH}}(\hat{\mu}_{ij}) &= var_{\text{LH}}(\hat{\mu}_{ij}) = var_{\text{SLH-ORG}}(\hat{\mu}_{ij}) = var_{\text{SLH-IND}}(\hat{\mu}_{ij}) \\ &\le var_{\text{IID}}(\hat{\mu}_{ij}). \end{aligned} \tag{3.17}$$

(ii) $$var_{\text{DSLH}}(\hat{\eta}_i) = var_{\text{SLH-IND}}(\hat{\eta}_i) \le var_{\text{LH}}(\hat{\eta}_i) \le var_{\text{IID}}(\hat{\eta}_i). \tag{3.18}$$

(iii) *For $\hat{\eta}$,*

$$var_{\text{DSLH}}(\hat{\eta}) \le var_{\text{SLH-IND}}(\hat{\eta}) \le var_{\text{LH}}(\hat{\eta}) \le var_{\text{IID}}(\hat{\eta}). \tag{3.19}$$

By dropping the monotonicity assumptions in Theorem 1, we have a more general result for $\hat{\mu}_{ij}$, $\hat{\eta}_i$ and $\hat{\eta}$ under doubly sliced Latin hypercube sampling.

Theorem 2. *Suppose that the $E[f_{ij}(\boldsymbol{x})]^2$ are well defined and finite. Let f_{ij}^{-k} be the functional main effect for the variable x_k of $\boldsymbol{x} = (x_1, \ldots, x_d)$ in the ANOVA decomposition of f_{ij} at (3.6). Let m, s, s_1 and s_2 be positive integers with $n = ms$ and $s = s_1 s_2$. For the six schemes of Definition 1, as $n \to \infty$ with s_1 and s_2 fixed, we have the following*

(i) *For $i = 1, \ldots, s_2$, $j = 1, \ldots, s_1$, and $\hat{\mu}_{ij}$ based on the slice $\boldsymbol{D}_{ij}$,*

$$\begin{aligned} \text{var}_{\text{DSLH}}(\hat{\mu}_{ij}) &= \text{var}_{\text{SLH-ORG}}(\hat{\mu}_{ij}) = \text{var}_{\text{SLH-IND}}(\hat{\mu}_{ij}) = \text{var}_{\text{LH}}(\hat{\mu}_{ij}) \\ &= \sigma_{ij}^2 \frac{s_1 s_2}{n} - \frac{s_1 s_2}{n} \sum_{k=1}^{d} \int_0^1 [f_{ij}^{-k}(x_k)]^2 dx_k + o(n^{-1}). \end{aligned} \tag{3.20}$$

(ii) *For $\hat{\eta}_i$ associated with the slices $\boldsymbol{D}_i = \bigcup_{j=1}^{s_1} \boldsymbol{D}_{ij}$,*

$$\begin{aligned} \text{var}_{\text{DSLH}}(\hat{\eta}_i) &= \text{var}_{\text{SLH-IND}}(\hat{\eta}_i) \\ &= \frac{s_1 s_2}{n} \sum_{j=1}^{s_1} \lambda_{ij}^2 \sigma_{ij}^2 - \frac{s_1 s_2}{n} \sum_{j=1}^{s_1} \Big\{ \lambda_{ij}^2 \sum_{k=1}^{d} \int_0^1 [f_{ij}^{-k}(x_k)]^2 dx_k \Big\} \\ &\quad + o(n^{-1}). \end{aligned} \tag{3.21}$$

(iii) *For $\hat{\eta}$ associated with all $\boldsymbol{D}_{ij}$s,*

$$\begin{aligned} \text{var}_{\text{DSLH}}(\hat{\eta}) = \frac{s_1 s_2}{n} \sum_{i=1}^{s_2} \sum_{j=1}^{s_1} \lambda_{ij}^2 \sigma_{ij}^2 - \frac{s_1 s_2}{n} \sum_{i=1}^{s_2} \sum_{j=1}^{s_1} \Big\{ \lambda_{ij}^2 \sum_{k=1}^{d} \int_0^1 [f_{ij}^{-k}(x_k)]^2 dx_k \Big\} \\ + o(n^{-1}). \end{aligned} \tag{3.22}$$

In (i), (ii) and (iii) of the theorem, the main effects $f_{ij}^{-k}(x_k)$ are filtered out, thus achieving variance reduction similar to an ordinary LHD. When all the f_{ij}s are the same and $\lambda_{ij} = (s_1 s_2)^{-1}$, we have $\sigma^2 = \sigma_{ij}^2 = \text{var}[f(\boldsymbol{x})]$ and (iii)

Table 1. RMSEs of $\hat{\mu}_{11}$ in Example 3.

	DSLH	SLH-ORG	SLH-IND	SLH-SPL
$m = 5$	0.4671	0.4756	0.4597	0.8002
$m = 10$	0.2310	0.2306	0.2337	0.5215
$m = 20$	0.1154	0.1134	0.1192	0.3646
$m = 40$	0.0576	0.0584	0.0608	0.2527

Table 2. RMSEs of $\hat{\eta}_1$ in Example 3.

	DSLH	SLH-ORG	SLH-IND	SLH-SPL
$m = 5$	0.1177	0.1403	0.1137	0.1183
$m = 10$	0.0579	0.0720	0.0586	0.0585
$m = 20$	0.0284	0.0355	0.0298	0.0299
$m = 40$	0.0150	0.0177	0.0147	0.0145

Table 3. RMSEs of $\hat{\eta}$ in Example 3.

	DSLH	SLH-ORG	SLH-IND	SLH-SPL
$m = 5$	0.1140	0.1143	0.1640	0.1203
$m = 10$	0.0570	0.0579	0.0811	0.0574
$m = 20$	0.0296	0.0288	0.0412	0.0295
$m = 40$	0.0146	0.0145	0.0210	0.0149

reduces to $\mathrm{var}_{DSLH}(\hat{\eta}) = \sigma^2/n - (1/n)\sum_{k=1}^{d}\int_0^1 [f_{-k}(x_k)]^2 dx_k + o(n^{-1})$, which is similar to that of an SLHD of n runs as given in Qian (2012) and that of an ordinary LHD of n runs as given in Stein (1987) and Loh (1996). For the case of different functions, the LH scheme, the SLH scheme, and the DSLH scheme are asymptotically equivalent in the sense that $\mathrm{var}_{LH}(\hat{\eta})$, $\mathrm{var}_{SLH}(\hat{\eta})$ (defined in Qian (2012)), and $\mathrm{var}_{DSLH}(\hat{\eta})$ are all $O(n^{-1})$.

4. Numerical Illustration

In this section, we provide numerical experiments to illustrate some theoretical results in Section 3. We focus on the comparison between DSLH and the three schemes related to SLH since it was illustrated in Qian (2012) that SLH outperforms LH and IID as a sampling method.

Example 3. This example uses the five-dimensional function (Drew and Homem-de Mello (2005))

$$f(\boldsymbol{x}) = \log(x_1x_2x_3x_4x_5), \tag{4.1}$$

where $\boldsymbol{x} = (x_1, x_2, x_3, x_4, x_5)$ is uniformly distributed on $[0, 1]^5$. Suppose we want to run this model in a batch sequential mode with a fixed batch size. The objective is to estimate the mean of the output given the distribution of the inputs. Suppose we would like to run the experiments in four batches with a

Table 4. RMSEs of $\hat{\mu}_{11}$ in Example 4.

	DSLH	SLH-ORG	SLH-IND	SLH-SPL
$m = 5$	0.1165	0.1149	0.1161	0.1581
$m = 10$	0.0629	0.0577	0.0620	0.1055
$m = 20$	0.0326	0.0338	0.0329	0.0710
$m = 40$	0.0184	0.0184	0.0178	0.0495

fixed batch size m. We are interested in the statistical properties of the sample average of the output after running one batch, two batches, and four batches. Let μ be the true mean of the function. Let $\hat{\mu}_{11}$, $\hat{\mu}_{12}$, $\hat{\mu}_{21}$ and $\hat{\mu}_{22}$ be the sample average of the output for the first batch, the second batch, the third batch and the fourth batch respectively. Let $\lambda_{11} = \lambda_{12} = \lambda_{21} = \lambda_{22} = 0.25$. Therefore, we are interested in the statistical properties of $\hat{\mu}_{11}$ and the quantities

$$\begin{aligned} \hat{\eta}_1 &= \lambda_{11}\hat{\mu}_{11} + \lambda_{12}\hat{\mu}_{12}, \\ \hat{\eta} &= \lambda_{11}\hat{\mu}_{11} + \lambda_{12}\hat{\mu}_{12} + \lambda_{21}\hat{\mu}_{21} + \lambda_{22}\hat{\mu}_{22}. \end{aligned}$$

We compare DSLH, SLH-ORG, SLH-IND, and SLH-SPL for this purpose. For DSLH, we generated a DSLHD$(m, 2, 2, 5)$ and assigned $\boldsymbol{D}_{11}$ as the first batch, $\boldsymbol{D}_{12}$ as the second batch, $\boldsymbol{D}_{21}$ as the third batch, $\boldsymbol{D}_{22}$ as the fourth batch. For SLH-ORG, we generated an SLHD of four slices, each with size m and then treated the four slices as four batches. For SLH-IND, we independently generated two SLHDs, each with two slices of slice size m. We then treated the first slice in the first SLHD as the first batch, the second slice in the first SLHD as the second batch, the first slice in the second SLHD as the third batch and the second slice in the second SLHD as the fourth batch. For SLH-SPL, we generated an SLHD of two slices, each with size $2m$. We then randomly split the first slice into two subsets of m runs and used the first subset as the first batch and the second subset as the second batch. Similarly we randomly split the second slice into two subsets of m runs and use the first subset as the third batch and the second subset as the fourth batch.

For each scheme, we computed $\hat{\mu}_{11}$, $\hat{\eta}_1$ and $\hat{\eta}$ for $m = 5, 10, 20, 40$. This was replicated 2,000 times. Tables 1, 2, and 3 present the root mean square error (RMSE) of $\hat{\mu}_{11}$, $\hat{\eta}_1$, and $\hat{\eta}$ over these 2,000 replications. They clearly show that, for each value of m, the DSLH scheme achieves the most variance reduction. However, the three SLH schemes all have their drawbacks. Specifically, $\hat{\mu}_{11}$ under the SLH-SPL scheme has a significantly larger variance than that under the other schemes; $\hat{\eta}_1$ under the SLH-ORG scheme has a significantly larger variance than the other schemes; $\hat{\eta}$ under the SLH-IND scheme has a significantly larger variance than the other schemes.

Table 5. RMSEs of $\hat{\eta}_1$ in Example 4.

	DSLH	SLH-ORG	SLH-IND	SLH-SPL
$m = 5$	0.0320	0.0367	0.0320	0.0300
$m = 10$	0.0171	0.0185	0.0166	0.0173
$m = 20$	0.0092	0.0104	0.0091	0.0089
$m = 40$	0.0050	0.0060	0.0051	0.0049

Table 6. RMSEs of $\hat{\eta}_2$ in Example 4.

	DSLH	SLH-ORG	SLH-IND	SLH-SPL
$m = 5$	0.0287	0.0359	0.0318	0.0312
$m = 10$	0.0158	0.0197	0.0165	0.0169
$m = 20$	0.0090	0.0102	0.0094	0.0092
$m = 40$	0.0051	0.0057	0.0053	0.0052

Table 7. RMSEs of $\hat{\eta}$ in Example 4.

	DSLH	SLH-ORG	SLH-IND	SLH-SPL
$m = 5$	0.0337	0.0344	0.0454	0.0331
$m = 10$	0.0181	0.0186	0.0235	0.0188
$m = 20$	0.0103	0.0103	0.0132	0.0105
$m = 40$	0.0059	0.0059	0.0073	0.0060

Table 8. RMSEs of $\hat{\eta}_1$ in Example 4 with different λ's.

	DSLH	SLH-ORG	SLH-IND	SLH-SPL
$m = 5$	0.0344	0.0372	0.0352	0.0386
$m = 10$	0.0187	0.0211	0.0178	0.0237
$m = 20$	0.0102	0.0115	0.0099	0.0148
$m = 40$	0.0056	0.0064	0.0059	0.0096

Table 9. RMSEs of $\hat{\eta}_2$ in Example 4 with different λ's.

	DSLH	SLH-ORG	SLH-IND	SLH-SPL
$m = 5$	0.0357	0.0389	0.0351	0.0399
$m = 10$	0.0185	0.0201	0.0182	0.0233
$m = 20$	0.0105	0.0111	0.0107	0.0153
$m = 40$	0.0057	0.0063	0.0057	0.0100

Table 10. RMSEs of $\hat{\eta}$ in Example 4 with different λ's.

	DSLH	SLH-ORG	SLH-IND	SLH-SPL
$m = 5$	0.0399	0.0396	0.0500	0.0483
$m = 10$	0.0221	0.0213	0.0259	0.0295
$m = 20$	0.0122	0.0119	0.0148	0.0197
$m = 40$	0.0070	0.0070	0.0083	0.0131

Example 4. This example uses the four functions $f_{11}(\boldsymbol{x}) = \log(1/\sqrt{x_1}+1/\sqrt{x_2})$, $f_{12}(\boldsymbol{x}) = \log(0.98/\sqrt{x_1}+0.95/\sqrt{x_2})$, $f_{21}(\boldsymbol{x}) = \log(1.02/\sqrt{x_1}+1.02/\sqrt{x_2})$, $f_{22}(\boldsymbol{x}) = \log(1/\sqrt{x_1}+1.03/\sqrt{x_2})$. Suppose f_{11} and f_{12} are two variants of a first computer model, and f_{21} and f_{22} are two variants of a second. Let $\mu_{11}, \mu_{12}, \mu_{21}, \mu_{22}$ denote the mean of $f_{11}, f_{12}, f_{21}, f_{22}$, respectively. We are interested in the mean of each variant and the quantities $\eta = (\mu_{11}+\mu_{12}+\mu_{21}+\mu_{22})/4$, $\eta_1 = (\mu_{11}+\mu_{12})/2$, $\eta_2 = (\mu_{21}+\mu_{22})/2$. We compared SLH-ORG, SLH-IND, SLH-SPL and DSLH using the setup of Example 3. The RMSEs of $\hat{\mu}_{11}$, $\hat{\eta}_1$, $\hat{\eta}_2$ and $\hat{\eta}$ are shown in Tables 4−7, respectively. The main conclusion is that the DSLH scheme works well for estimating all four parameters, while SLH-ORG does not estimate η_1 and η_2 as efficiently, SLH-IND does not estimate η as efficiently, and SLH-SPL does not estimate μ_{ij} as efficiently, $i, j = 1, 2$.

The results in Example 4 are for equal λ's. But others can be of interest. For illustration purposes, we let $\lambda_{11} = 1/3$, $\lambda_{12} = 1/6$, $\lambda_{21} = 1/3$ and $\lambda_{22} = 1/6$. The simulation was then run with the same setup as for equal λ's. The RMSEs are shown in Tables 8−10. We do not show the results for μ_{ij}'s as they are exactly the same as before. While the same conclusions emerge, the advantage of DSLH over SLH-SPL is more pronounced in the case of different λ's; SLH-SPL cannot guarantee the design for each function is an LHD, while DSLH can.

5. Discussion

We have proposed general sliced Latin hypercube designs that have multiple layers, at each of which there are multiple LHDs that can be further sliced into even smaller LHDs at the next layer. An ordinary LHD and an SLHD (Qian (2012)) are general sliced Latin hypercube designs with zero layer and one layer, respectively. A special case of general sliced Latin hypercube designs with two layers, doubly sliced Latin hypercube design (DSLHD), is studied in detail. Potential applications of DSLHD, other than computer experiments, include cross-validation, stochastic optimization, and tuning parameter selection in smoothing splines or similar nonparametric regression methods.

In numerical experiments DSLHD outperforms SLHD for collective evaluation of computer models when we have more than one computer model and each has the same number of variants. DSLHD also provides more flexibility for batch sequential estimation of the mean of a single computer model. These advantages stem from the more flexible structure of DSLHD.

One research direction is to apply the sliced structure to construct designs for computer experiments with both quantitative and qualitative factors. The number of layers can depend on the number of qualitative factors. Ideally they should be the same. See Qian, Wu, and Wu (2008) for details of Gaussian process modeling for both quantitative and qualitative factors.

Another interesting research direction is to obtain a central limit theorem for doubly sliced Latin hypercube sampling. Central limit theorems for Latin hypercube sampling have been given in simpler situations (Stein (1987); Owen (1992); Loh (1996)). The central limit theorem for doubly sliced Latin hypercube sampling is technically more involved because of the complicated covariance structure among the sampled points.

Acknowledgements

Xie and Wu's research was supported by two NSF grants. Xiong's research was supported by the National Natural Science Foundation of China (Grant No. 11271355). Qian's research was supported by an NSF Career Award (DMS 1055214).

References

Drew, S. and Homem-de Mello, T. (2005). Some large deviation results for Latin hypercube sampling. *Proceedings of the* 2005 *Winter Simulation Conference*, 674-681.

Fang, K. T., Li, R. Z. and Sudjianto, A. (2005). *Design and Modeling for Computer Experiments.* Chapman Hall/CRC Press, New York.

Lehmann, E. L. (1966). Some concepts of dependence. *Ann. Math. Statist.* **37**, 1137-1153.

Loh, W. L. (1996). On Latin hypercube sampling. *Ann. Statist.* **24**, 2058-2080.

McKay, M. D., Conover, W. J. and Beckman, R. 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.

Owen, A. B. (1992). A central limit theorem for Latin hypercube sampling. *J. Roy. Statist. Soc. Ser. B* **54**, 541-551.

Qian, P. Z. G. (2009). Nested Latin hypercube sampling. *Biometrika* **96**, 957-970.

Qian, P. Z. G. (2012). Sliced Latin hypercube sampling. *J. Amer. Statist. Assoc.* in press.

Qian, P. Z. G., Wu, H. and Wu, C. F. J. (2008). Gaussian process models for computer experiments with qualitative and quantitative factors. *Technometrics* **50**, 383-396.

Santner, T. J., Williams, B. J. and Notz, W. I. (2003). *The Design and Analysis of Computer Experiments.* Springer, New York.

Stein, M. (1987). Large sample properties of simulations using Latin hypercube sampling. *Technometrics* **29**, 143-151.

Tang, B. (1993). Orthogonal array-based Latin hypercubes. *J. Amer. Statist. Assoc.* **88**, 1392-1397.

Tuo, R., Wu, C. F. J. and Yu, D. (2013). Modeling of computer experiments with different mesh densities. *Technometrics* in press.

Williams, B., Morris, M. and Santner, T. (2009). Using multiple computer models/multiple data sources simultaneously to infer calibration parameters. Presentation, The 2009 INFORMS Annual Conference, San Diego, CA.

Wu, C. F. J and Ding, Y. (1998). Construction of response surface designs for qualitative and quantitative factors. *J. Statist. Plann. Inference* **71**, 331-348.

Science and Algorithms, Netflix, Los Gatos, CA 95032, U.S.A.

E-mail: kxie@netflix.com

Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China.

E-mail: xiong@amss.ac.cn

Department of Statistics, University of Wisconsin-Madison, Madison, WI 53706, U.S.A.

E-mail: peterq@stat.wisc.edu

School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332, U.S.A.

E-mail: jeffwu@isye.gatech.edu

(Received November 2012; accepted August 2013)