Received: 6 October 2017 | Revised: 19 February 2019 | Accepted: 24 March 2019

DOI: 10.1002/qre.2485

RESEARCH ARTICLE

WILEY

Sequential latin hypercube design with both space-filling and projective properties

XiaoJian Zhou[1,2] | Dennis K. J. Lin[2] | XueLong Hu[1] | Linhan Ouyang[3]

[1]School of Management, Nanjing University of Posts and Telecommunications, Nanjing, China

[2]Department of Statistics, Pennsylvania State University, State College, Pennsylvania

[3]College of Economics and Management, Nanjing University of Aeronautics and Astronautics, Nanjing, China

Correspondence
XiaoJian Zhou, School of Management, Nanjing University of Posts and Telecommunications, Nanjing 210003, China.
Email: xjzhou@njupt.edu.cn

Funding information
National Natural Science Foundation of China, Grant/Award Number: 71401080, 71702072, 71802110 and 71872088; National Security Agent, Grant/Award Number: H98230-15-1-0253; Natural Science Foundation of Jiangsu, Grant/Award Number: BK20170894; Social Science Foundation of Jiangsu, Grant/Award Number: 17GLB016; Social Science Foundation of NJUPT, Grant/Award Number: 1311 Talent Fund , NY218064, NYJD217006 and NYS216011; State Scholarship Fund of China, Grant/Award Number: 201508320059

Abstract

Space-filling and projective properties are probably the two most important features in computer experiment. The existing research works have tried to develop different kinds of sequential Latin hypercube design (LHD) to meet these two properties. However, most if not all of them cannot simultaneously ensure these two properties in their versions of sequential LHD. In this paper, we propose a novel sequential LHD that can simultaneously meet the space-filling and the projective properties at each stage. A search algorithm is employed to find how many design points should be added in each stage to ensure the projective property; and the "Maximin" criterion is used to meet the space-filling property. Two kinds of examples for low dimension and higher dimension are presented to illustrate how these sequential sampling processes are realized. The proposed method can be applied to the areas where computationally expensive simulations are involved.

KEYWORDS

computer experiment, Latin hypercube design, metamodel, sequential sampling

1 | INTRODUCTION

Design and analysis of computer experiment (DACE) is widely utilized in expensive computer simulations. In order to deal with the computational burden, the surrogate model (or metamodel) is used to replace the computationally expensive simulations. Whether we can obtain a reliable metamodel strongly depends on what design points we have used to construct it. It is widely believed that a design of computer experiment should possess good space-filling and projective properties, which can be traced in the work of Johnson et al, Jin et al, Yang et al, Chen et al, Wang et al, Steinberg and Lin, Yin et al,[1-7] and Yang et al.[8] Space-filling means the design points should fill up the design space as much as possible. Projective property requires that the projections of design points onto each dimension should be evenly located. That is, there are no replication and point clustering in each interval of every design dimension. To guarantee these two properties, Maximin Latin hypercube design (LHD) was proposed in Johnson et al.[1] After that, a great deal of literature proposed different kinds of LHDs with good space-filling and projective properties. However, they are computationally inefficient and time consuming in high-dimension problems. For instance, Kenny et al[9] used the columnwise-pairwise (CP) algorithm[10] for constructing optimal symmetrical LHDs.[9] reported that generating an optimal 25 × 4 LHDs using CP could take several hours on a Sun SPARC 20 workstation. For a design as large as 100 × 10, the computational cost could be formidable; thus, search processes often stopped before finding a good design. In addition, all these methods construct the design points at once, meaning they are all one-stage sampling approaches.

To reduce the computational cost, sequential LHD was proposed to gradually increase the design points. The benefit of sequential LHD is that one can sequentially generate design points until certain accuracy criterion is satisfied. On the basis of the integrated mean squared error (IMSE) criterion, Sacks et al[11] proposed a sequential sampling algorithm to update the kriging metamodel. Jin et al[12] used cross-validation (CV) error to sequentially generate new design points. Kim et al[13] developed a maximum modified CV error criterion to generate new-added design points. Alternatively, other kinds of sequential strategy mainly focus on the space-filling or/and projective properties of LHD. Considering the effects of the existing points, an inheriting (ie, historical design experiments can be inherited in later iterations) LHD approach was presented by Wang[14] to construct new LHD points in reduced design spaces and obtain the properties of LHD in a subspace. Using a genetic algorithm optimization process, a Quasi-LHD method was developed by Xiong et al[15] to achieve space-filling and approximate projective properties. Employing a Monte Carlo approach, Crombecq et al[16] proposed a mc-intersite-proj-th (where, mc: Monte Carlo; proj: projected; th: threshold) approach to generate design points sequentially. Combining Monte Carlo method and space-reduction strategy (MCSR), Liu et al[17] proposed a Maximin LHD approach to improve space-filling property. Taking the advantage of a kind of successive local enumeration (SLE) approach, Zhu et al and Long et al[18,19] used a sequential-successive local enumeration (S-SLE) method to develop a sequential Maximin LHD method.

There are two points on sequential sampling strategy needed to emphasize:

1. Should we sequentially extract design points according to the "accuracy" of metamodel? In the whole process of sequential sampling, design points are often extracted in the areas where the accuracy of a metamodel is enhanced just with respect to certain criterion.[11-13] used IMSE or CV to generate new design points to update the metamodel. However, using CV error (or IMSE) criterion to choose metamodel is not reliable (see, for example, Zhou et al, Zhou and Jiang, and Ouyang et al[20-23]). A metamodel with a small CV/IMSE error does not imply that it has a good prediction capacity. Therefore, the CV/IMSE error may not be appropriate to determine the design points in the sequential sampling process. For instance, a metamodel (such as Kriging, support vector machine [SVR],[24-26] radial basis function [RBF]) has a small CV/IMSE value for a specific set of samples, but it maybe has an unsatisfactory prediction result in the whole design space. To overcome this drawback, we choose to directly guarantee the space-filling and projective properties in the whole sequential sampling process (or at least in the first several stages of the whole sequential process). So a new kind of sequential LHD method is proposed in this paper to guarantee these two properties simultaneously in the whole sequential sampling process. More importantly, the proposed method does not depend on the accuracy of a specified metamodel, so it is metamodel independent. Note that (a) in the first several stages, only a certain percent (less than 100%) of the previous samples are needed in the current stage to ensure the projective properties. For example, if there are six samples in the previous stage, then perhaps, only two samples are needed to add to guarantee the whole set of these eight $(6 + 2)$ samples is an LHD. (b) In the latter stages, when the number of samples grows up, maybe 100% of the previous samples are needed. For example, if there are 30 samples in the previous stage, then perhaps, another 30 samples are needed to add to guarantee the whole set of these 60 $(30 + 30)$ samples is an LHD. Double number of samples of the previous stage can definitely guarantee the whole set of the samples in the current stage to be a LHD. In other words, if the number of samples in the previous stage is N, then add another N samples in the current stage can always ensure the whole set of these 2N samples is an LHD. We explain it furthermore: if we split every dimension of the design space into 2N parts and project the original N samples into these dimensions, then there are definitely N vacant intervals (vacant interval means there is no projection within it) on each dimension. Therefore, another N samples can be added to the original N samples, such that every vacant interval has only one projection. Therefore, the set of the total 2N samples will always be an LHD.
2. Although some existing sequential sampling strategies also try to improve the space-filling and projective properties of LHD in the sequential sampling process, none of them can ensure the projective property at each stage. They can only obtain an "approximate" projective property (see, Xiong et al, Liu et al, and Long et al[15-17,19]). Some of them is only able to have local properties of LHD rather than those of global LHD; that is, it is an LHD in a local subspace

but not in the entire design space (see, Wang[14]). There, they can only guarantee the set of added if samples alone is an LHD in subspaces, but the whole set of samples (including the original samples and the new-added samples) is no longer an LHD in the whole space. Nevertheless, the proposed method in this paper can simultaneously ensure both the space-filling and the projective properties in the whole space at each stage of the whole sequential sampling process.

In a whole, the existing sequential LHD methods cannot ensure the projective property for the whole sequential process. For example, in the work of Xiong et al,[15] the authors use an optimization formulation to get the sequential samples, but it cannot ensure accurate projective property in each stage (please see the fourth figure in their article). Another example (please see the work of Wang[14]), where it is an LHD in a subspace, but it is no longer an LHD in the whole space, please see the sixth figure in their work.

In this paper, a search algorithm is used to determine how many design points should be taken to guarantee the projective property at each stage. The Maximin criterion is employed to ensure the space-filling property.

This paper is organized as follows. Section 2 details the proposed sequential LHD with an algorithm. Section 3 and section 4 present some examples about how to sequentially generate design points to meet these two basic properties of LHD. In section 5, a conclusion is given.

2 | PROPOSED SEQUENTIAL LHD

In order to guarantee the space-filling and projective properties at each stage of sequential LHD, we design the following algorithm. This algorithm is realized based on the version R2014a of MATLAB.

Algorithm 1 solves the following problems: (a) it determines how many design points should be employed in the next stage so that the whole set of the design points (including the original design points in the previous stage and the new-added design points in the current stage) possesses the projective property; (b) it finds the candidate locations (in one-dimensional design space, it refers to an interval; in two-dimensional design space, it refers to a plane; in three-dimensional design space, it refers to a cubical subspace; in higher dimensional design space, it refers to a hyper subspace), where the design points can be generated so that the projective property can be met; and (c) it generates the design points in the candidate locations according to Maximin criterion so that the space-filling property can be guaranteed.

Algorithm 1 (Sequential LHD):

Step 1: use the standard function *lhsdesign* in MATLAB to generate the initial design points with the number being $N_{initial}$.

Step 2: based upon the existing design points (denoted by S_{exist} with the number being N_{exist}), do the following steps. (Note: **find N_{add} using steps from 3 to 6*)

Step 3: let $k = 1$.

Step 4: separate every dimension of the design space into $N_{exist} + k$ intervals.

Step 5: judge whether each of those $(N_{exist} + k)$ intervals on every dimensions of the design space has no more than one projection point coming from the design points. If all of these intervals meet the above-mentioned requirement of having no more than one projection point, go to step 6; otherwise, let $k = k + 1$ and return to step 4. (In this step, the requirement is "having no more than one projection point" in each interval, ie, either "there is only one projection point 'or' there is no projection point" in each interval. When every dimension simultaneously meets this requirement implies that every dimension has k vacant interval[s]. This also implies we need to generate k sample[s] to meet the projective property.)

Step 6: let $N_{add} = k$ and the total set of samples with the number being $N_{total} = N_{exist} + N_{add}$ guarantees to form an LHD. (Note: *find the vacant intervals for each dimension using steps 7 and 8) (In the above-mentioned steps, we just find out how many vacant intervals lie on every dimension, but we still have not marked them; in the following two steps, we will do that work.)

Step 7: For $i = 1 : d$ (d is the number of the dimensions in the design space), do the followings: For the ith dimension, separate this dimension evenly into N_{total} intervals, and for each interval, judge whether one projection of these N_{exist} samples $S_j = [x_{j1} \dots x_{ji} \dots x_{jd}]$ $(j = 1, \dots, N_{exist})$ is located in this interval; in other words, judge which intervals have projections. The N_{exist} samples can be expressed as follows:

$$\begin{bmatrix} x_{11} & \cdots & x_{1i} & \cdots & x_{1d} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{N_{exist}1} & \cdots & x_{N_{exist}i} & \cdots & x_{N_{exist}d} \end{bmatrix},$$

and the projection into the ith dimension of these N_{exist} samples is

$$\begin{bmatrix} x_{1i} \\ \vdots \\ x_{N_{exist}i} \end{bmatrix}$$

The purpose of this step is to find out the vacant intervals for the ith dimension according to the ith projection $\left[x_{1i} \;\cdots\; x_{N_{exist}i}\right]^T$. Here, a subloop is needed to do such judgement (which means for each of the intervals in all of the dimensions, to judge whether it is vacant), in other words, for each of these intervals, all of the members of the ith projection $\left[x_{1i} \;\cdots\; x_{N_{exist}i}\right]^T$ are used to judge whether one of them locates in this interval, if not, mark it as vacant interval.

(Note: *generate the best samples conforming the Maximin criterion by using steps from 8 to 12]

Step 8: based upon the vacant intervals on every dimension found in the step 8, N_{add} design points can be randomly generated (denoted by S_{best}). This randomly generating process is done as follows.

Randomly generate N_{add} numbers $(x_{1i}, \ldots, x_{N_{add}i})$ (where $x_{ki} \in [0,1](k = 1, \ldots, N_{add})$) in the N_{add} vacant intervals for ith dimension and do so for every dimension of the design space (d dimensions), then the N_{add} samples can be expressed as

$$\begin{bmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{N_{add}1} & \cdots & x_{N_{add}d} \end{bmatrix}.$$

Next, run a random permutation for the N_{add} numbers in each dimension, then the final randomly generated samples can be obtained. When these design points are projected into each dimension, there is only one point within each interval. The existing design points (denoted by S_{exist}) and the new-added design points (S_{best}) are put together (let $S_{total} = S_{exist} \cup S_{best}$); and the minimum distance in S_{total} is calculated to be D_{best}.

Step 9: generate a new set of design points (denoted by S_{new}) in above-mentioned vacant subspaces (if in one-dimensional design space, they are intervals), calculate the minimum distance in $S_{exist} \cup S_{new}$, and denote this minimum distance by D_{new}. If $D_{new} > D_{best}$, let $S_{best} = S_{new}$.

Step 10: judge whether the number of the iterations meets the preset upper limit (denoted by *Maxiter*). If so, go to step 11; otherwise, return to step 9.

Step 11: get the best set of design points S_{best}, which has the maximum value among all these minimum distances.

Step 12: judge whether this sequential sampling process should be continued; if so, return to step 2; otherwise, end this sequential sampling process and obtain the final design points.

The theoretical basis on determining the number N_{add} in the proposed algorithm is given in Theorem 1 below.

Theorem 1. *Given design points $(x_{ij})_{n\times d}$ (normalized into the hyperspace $[0,1]^d$) of the previous stage, if any two elements in the set $\left\{\lfloor (n+k)x_{ij} \rfloor, i = 1, \ldots, n\right\}$ are not equal for any dimension j ($j = 1, \ldots, d$), then k ($k \leqslant n$) additional design points can be added at the current stage so that the design points $(x_{ij})_{(n+k)\times d}$ are an LHD.*

The proof of Theorem 1 is presented in the APPENDIX.

Remark 1. Theorem 1 shows that one can find a proper number k to guarantee the projective property at each stage of the sequential sampling process. After finding the number k, the steps from 9 to 12 in *Algorithm 1* can be used to guarantee the space-filling property. Therefore, the proposed method in this article can guarantee the space-filling as well as the projective properties.

3 | EXAMPLES IN LOW DIMENSION

In this section, two examples are given to illustrate how to sequentially generate design points at each stage, letting the whole set of design points at each stage meets the space-filling and projective properties. In order to clearly show the distribution of the new-added design points at each stage, we give a one-dimensional and a two-dimensional examples

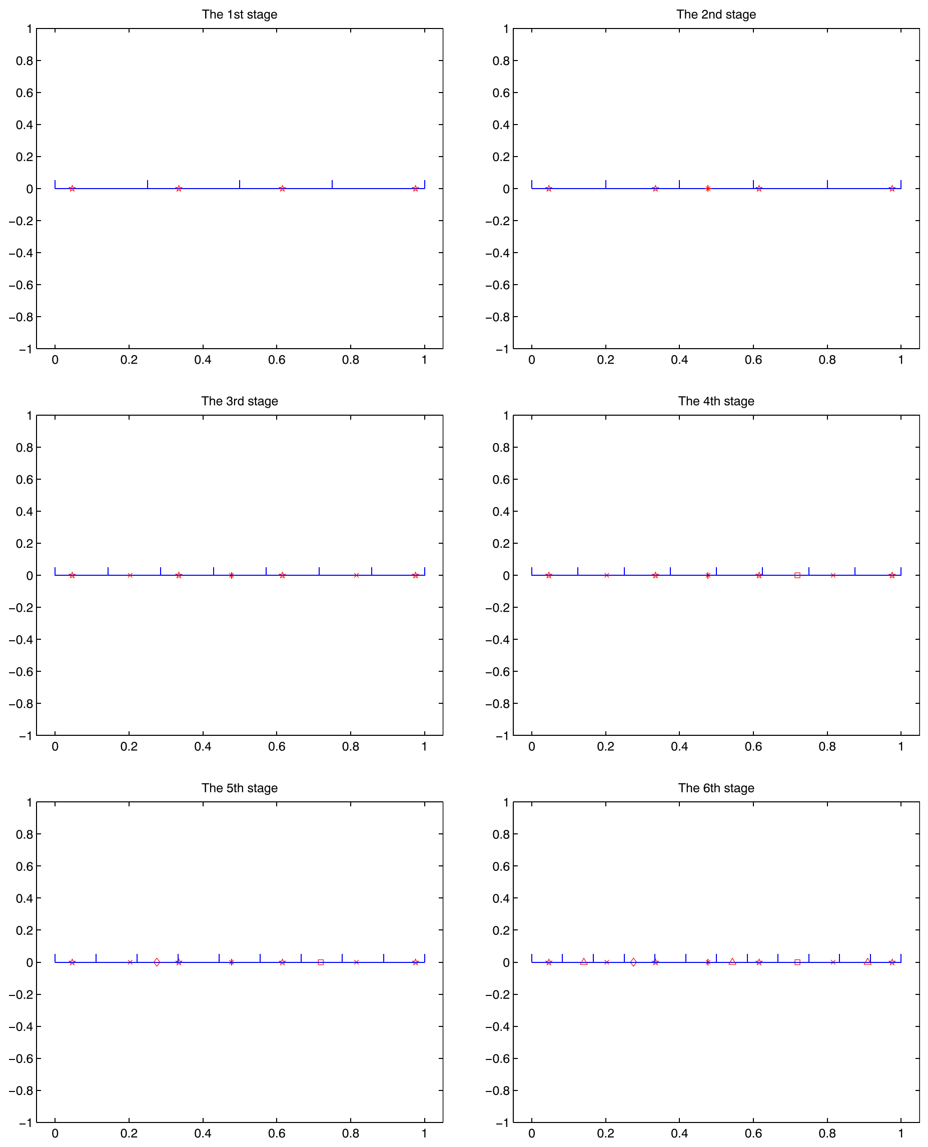

FIGURE 1 The distribution of the design points at each stage of the sequential process (1D) [Colour figure can be viewed at wileyonlinelibrary.com]

to illustrate the effectiveness of our method. Certainly, the proposed strategy also can be used to any higher dimensional problems, which will be illustrated in the next section.

3.1 | A one-dimensional example

A one-dimensional example is illustrated. The whole framework is presented in Algorithm 1. The number of the initial design points is set to be 4, and the numbers that should be added at each stage are determined by the steps from 3 to 6 in Algorithm 1. In order to guarantee the space-filling property, the steps from 8 to 11 in Algorithm 1 are utilized to generate the new-added design points in the vacant intervals at each stage. Figure 1 shows how the design points distribute at each stage of the sequential process. To distinguish the design points in different stages, we use dots with different kinds of shape to denote them. In the whole sequential process, the new-added design points are determined by the design points in the previous stage.

The detailed steps at each stage are presented as follows:

a. At the first stage, an initial LHD, including four samples ($N_{initial} = 4$), is generated, with the shape "$\star$" representing them.

b. At the second stage, the steps from 3 to 6 in Algorithm 1 are employed to find out how many vacant intervals exist on the design space (one dimension in this example) and get $N_{add} = 1$. Then the whole one-dimensional space can be separated into $N_{initial} + N_{add} = 4 + 1 = 5$ intervals, and we can easily observe from the second subfigure of Figure 1 that the third interval is vacant, and each of the remaining intervals has and only has one sample. Of course, this job to judge whether there is a design point in an interval is done by the step 7 in Algorithm 1 implemented by computer. Then, a sample can be randomly generated in this vacant interval. In order to get the space-filling property, we randomly generate a sample in the third vacant interval and repeat this process for Maxiter (Maxiter = 20 in this paper) times and finally get the best one conforming the Maximin criterion as what we want in this stage, using the shape "$*$" representing it.
c. At the third stage, also use the steps from 3 to 6 in Algorithm 1 to find out how many vacant intervals exist in the design space and get $N_{add} = 2$. Then the whole one-dimensional space can be separated into $N_{exist} + N_{add} = 5 + 2 = 7$ intervals, with two intervals (the second and sixth ones) are vacant, and each of the remaining intervals has one sample respectively. Then, two samples can be randomly generated into these two vacant intervals. Similar to the previous stage, we repeat this process for Maxiter times and get the best two samples as the final samples, with the shape "$\times$" representing them.
d. At the fourth stage, $N_{add} = 1$. The whole space can be separated into $N_{exist} + N_{add} = 7 + 1 = 8$ intervals; and we find that the 6th interval is vacant. So the sample can be randomly generated in this interval for Maxiter times and choose the best one meet the space-filling requirement as the final sample on this stage, with the shape "$\square$" representing it.
e. At the fifth stage, $N_{add} = 1$. The whole space can be separated into $N_{exist} + N_{add} = 8 + 1 = 9$ intervals; and vacant interval is the third one. A sample meeting the space-filling requirement is chosen and represented with the shape "$\diamond$."
f. At the sixth stage, $N_{add} = 3$. The whole one-dimensional space can be separated into $N_{exist} + N_{add} = 9 + 3 = 12$ intervals, with three intervals (the second, seventh, and 11th ones) are vacant, and each of the remaining intervals has one sample respectively. Then, three samples are randomly generated into these vacant intervals; and the Maximin criterion is employed to get the best samples, with the shape "$\triangle$" representing them. If the engineers want to do more stages, they can carry it out in a similar way.

3.2 | A two-dimensional example

Next, a two-dimensional example is employed. Similar to the one-dimensional example, four initial design points are generated using the standard function *lhsdesign* in MATLAB 2014a. The steps from 3 to 6 in Algorithm 1 are used to determine how many design points should be added in the next stage; and the steps from 8 to 11 in Algorithm 1 are employed to guarantee the space-filling property. The main framework is illustrated by Algorithm 1. The sequential samples-generating process is presented in Figure 2.

Similar to the stages in the one-dimensional example, we give the detailed description about how to add additional samples at each stage to form a LHD as follows:

a. At the first stage, four samples ($N_{initial} = 4$) are randomly generated to form a LHD, with the shape "+" representing them.
b. At the second stage, the steps from 3 to 6 in Algorithm 1 are employed to find out how many vacant intervals exist on every dimension and get $N_{add} = 1$. The horizontal axis thereby can be separated into $N_{initial} + N_{add} = 4 + 1 = 5$ intervals, with the fifth interval is vacant; and the vertical axis also can be separated into $N_{initial} + N_{add} = 4 + 1 = 5$ intervals, with the third interval (count it from the bottom of the design space) is vacant; then, only one subspace (two intervals can constitute one subspace in a two-dimensional design space) can be chosen to randomly generate a sample. Similar to that in the one-dimensional example, this randomly-generating process can be repeated for Maxiter times, and finally, the best one conforming the Maximin criterion is chosen as what we want in this stage to form an LHD. The final sample randomly generated in the vacant subspace is represented with the shape "$\star$."
c. At the third stage, $N_{add} = 1$. The horizontal axis can be separated into $N_{exist} + N_{add} = 5 + 1 = 6$ intervals, with the second interval is vacant; and the vertical axis also can be separated into $N_{exist} + N_{add} = 5 + 1 = 6$ intervals, with the third interval is vacant; then, the subspace (2,3) (the coordinate of this subspace) can be chosen to randomly generate a sample with the best Maximin, and the final sample is represented with the shape "$*$".

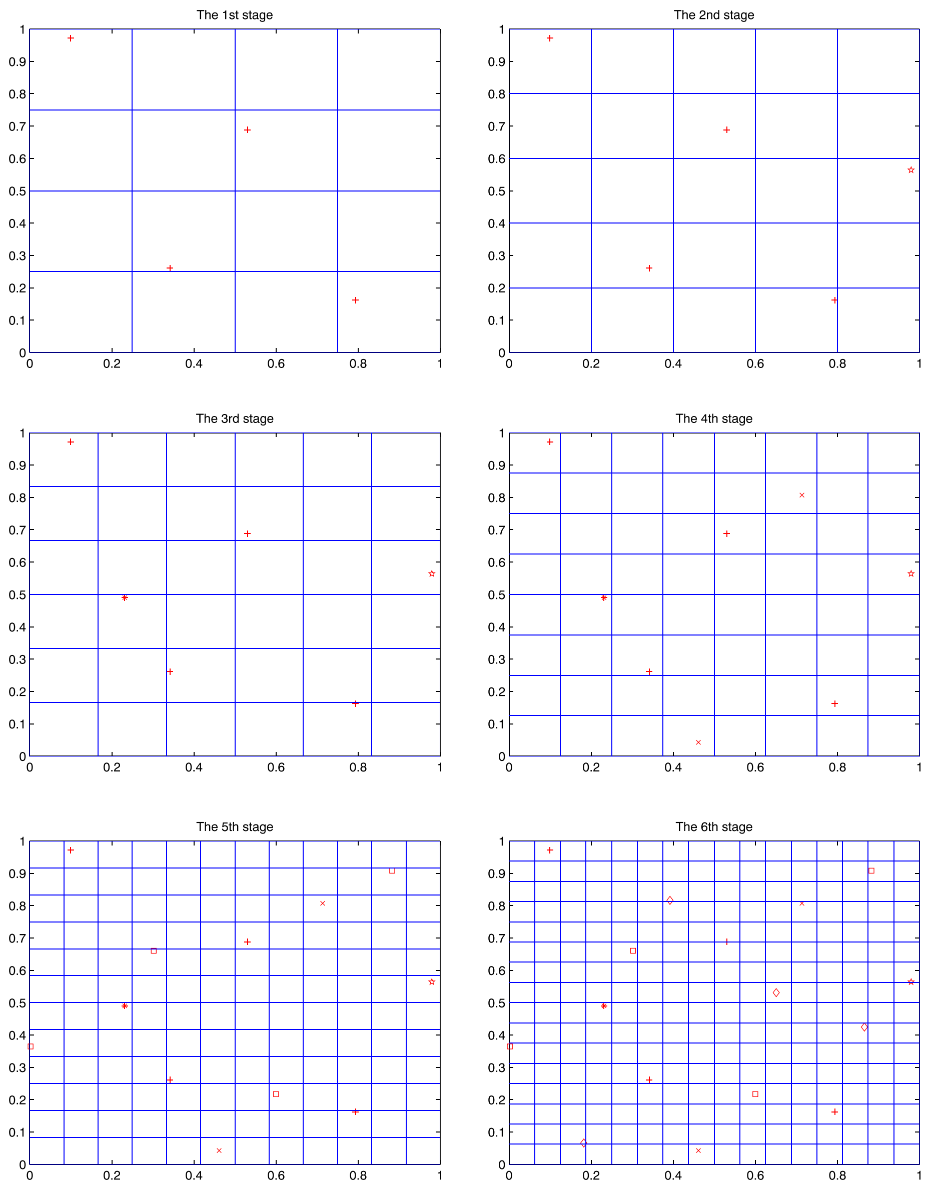

FIGURE 2 The distribution of the design points at each stage of the sequential process (2D) [Colour figure can be viewed at wileyonlinelibrary.com]

d. At the fourth stage, $N_{add} = 2$. The horizontal axis can be separated into $N_{exist} + N_{add} = 6 + 2 = 8$ intervals, with two intervals (the fourth and sixth ones) are vacant, and the vertical axis also can be separated into $N_{exist} + N_{add} = 6 + 2 = 8$ intervals, with two intervals (the first and seventh ones) are vacant. Then, there are four subspaces (4,1), (4,7), (6,1), (6,7) can be used to generate samples, but they have to meet the projection requirement of LHD, that is, there is only one projection point in one interval. Therefore, in order to generate two samples, there are two alternatives: one is the combination of subspace(4,1) and subspace (6,7), that means generating these two samples into these two subspaces; and the other is the combination of subspace (4,7) and subspace (6,1). In order to get the random samples, two random processes are used: the first random process is randomly choosing of one combination; and the second random process is randomly generating of two samples into the combination described above. These two random processes can be carried out simultaneously; and then, two samples are obtained. In order to get the space-filling property, the sample-generating process (including these two random processes mentioned above) can be repeated for Maxiter times, and the one with the best Maximin can be chosen as the final samples in this stage. The result is presented in the fourth subfigure of Figure 2, with the shape "×" representing them.

e. At the fifth stage, $N_{add} = 4$. The horizontal axis can be separated into $N_{exist} + N_{add} = 8 + 4 = 12$ intervals, with four intervals (the first, fourth, eighth, and 11th ones) are vacant, and the vertical axis also can be separated into

$N_{exist} + N_{add} = 8 + 4 = 12$ intervals, with four intervals (the third, fifth, eighth, and 11th ones) are vacant. Then there are $4! = 24$ combinations to get the subspaces that can be used to generate samples. One combination means a set of subspaces. For instance, (1,3), (4,5), (8,8), (11,11) is a combination where four samples can be randomly generated, and the projective property is met. Similar to the previous stage, we randomly get a combination and randomly generate samples in that combination. This process (including randomly getting a combination and randomly generating samples in the chosen combination) is repeated for Maxiter times and get the best samples that meet the space-filling property. The result is presented in the fifth subfigure of Figure 2 with the shape "□" representing them.

f. At the sixth stage, $N_{add} = 4$. Similar to stage 5, we also can find four "best" samples, letting them meet the projective and space-filling properties of LHD. The result is presented in the sixth subfigure of Figure 2, with the added samples described using the shape "⋄." If the engineers are pleased to do more stages, they can do it as above.

4 | EXAMPLE IN HIGHER DIMENSION

In this section, one five-dimensional example is presented to illustrate the effectiveness of the proposed method. Because the dimension of this example is larger than two, it is difficult to show intuitively as in low-dimensional examples. Therefore, we simply present data to describe the sequential process. In this part, four stages of the sequential process are described for this example. In this five-dimensional example, the generated data in each stage are shown to describe the sequential process.

a. At the first stage, four samples are randomly generated using the standard MATLAB function *lhsdesign*. The corresponding data, 4×5 LHD (four samples, five dimensions) are as follows:

$$\begin{bmatrix} 0.4482 & 0.3107 & 0.0979 & 0.3187 & 0.0277 \\ 0.2082 & 0.6744 & 0.5410 & 0.9456 & 0.5343 \\ 0.7267 & 0.9846 & 0.9063 & 0.0850 & 0.2718 \\ 0.8399 & 0.0571 & 0.2664 & 0.7276 & 0.9187 \end{bmatrix}.$$

b. At the second stage, the steps from 3 to 6 in Algorithm 1 are employed to find out how many vacant intervals exist on every dimension and get $N_{add} = 1$. For each vacant interval of each dimension, a real number can be generated within that interval. For instance, x_{i1} is randomly generated within the vacant interval of the first dimension; similarly, x_{i2}, x_{i3}, x_{i4}, and x_{i5} are randomly generated for the second, third, fourth, and fifth dimensions, respectively; combine them together, then the additional sample $[x_{i1}\, x_{i2}\, x_{i3}\, x_{i4}\, x_{i5}]$ is generated. This sample meets the projection requirement of LHD. If we want to get a space-filling LHD, we can repeat the above process for Maxiter times and get the best one conforming the Maximin criterion. In our case, the following samples are obtained:

$$\begin{bmatrix} 0.0372 & 0.5325 & 0.7230 & 0.4059 & 0.7774 \end{bmatrix},$$

so that the following total set of samples remains a LHD with projective and space-filling properties

$$\begin{bmatrix} 0.4482 & 0.3107 & 0.0979 & 0.3187 & 0.0277 \\ 0.2082 & 0.6744 & 0.5410 & 0.9456 & 0.5343 \\ 0.7267 & 0.9846 & 0.9063 & 0.0850 & 0.2718 \\ 0.8399 & 0.0571 & 0.2664 & 0.7276 & 0.9187 \\ 0.0372 & 0.5325 & 0.7230 & 0.4059 & 0.7774 \end{bmatrix}.$$

c. At the third stage, $N_{add} = 1$, and we can get another sample

$$\begin{bmatrix} 0.6167 & 0.4780 & 0.4875 & 0.5089 & 0.4051 \end{bmatrix}.$$

Similarly, combined with the existing samples, the following total set of samples still meets the requirements of LHD:

$$\begin{bmatrix} 0.4482 & 0.3107 & 0.0979 & 0.3187 & 0.0277 \\ 0.2082 & 0.6744 & 0.5410 & 0.9456 & 0.5343 \\ 0.7267 & 0.9846 & 0.9063 & 0.0850 & 0.2718 \\ 0.8399 & 0.0571 & 0.2664 & 0.7276 & 0.9187 \\ 0.0372 & 0.5325 & 0.7230 & 0.4059 & 0.7774 \\ 0.6167 & 0.4780 & 0.4875 & 0.5089 & 0.4051 \end{bmatrix}.$$

d. At the fourth stage, the steps from 3 to 6 in Algorithm 1 are used to find the vacant intervals for all of these dimensions, and $N_{add} = 2$, that is, two vacant intervals are found for every dimension. Similar to the previous stage, two design points can be randomly generated within those two vacant intervals. For instance, x_{i1} and $x_{i+1,1}$ are randomly generated within the vacant intervals of the first dimension; x_{i2} and $x_{i+1,2}$ are randomly generated for the second dimension; x_{i3} and $x_{i+1,3}$ are randomly generated for the third dimension; x_{i4} and $x_{i+1,4}$ are randomly generated for the fourth dimension; x_{i5} and $x_{i+1,5}$ are randomly generated for the fifth dimension; then, these two samples can be generated as follows:

$$\begin{bmatrix} x_{i1} & x_{i2} & x_{i3} & x_{i4} & x_{i5} \\ x_{i+1,1} & x_{i+1,2} & x_{i+1,3} & x_{i+1,4} & x_{i+1,5} \end{bmatrix}.$$

Furthermore, the positions of these two real numbers within each dimension also can be randomly permutated, for example:

$$\begin{bmatrix} x_{i1} & x_{i+1,2} & x_{i3} & x_{i+1,4} & x_{i5} \\ x_{i+1,1} & x_{i2} & x_{i+1,3} & x_{i4} & x_{i+1,5} \end{bmatrix}.$$

Therefore, there are $(N_{add}!)^{(d-1)}$ combinations. In this stage, there are $(2!)^{(5-1)} = 16$ combinations. Note that there are two kinds of random process involved: the first kind is that the numbers are randomly generated; and the second one is that the positions of these real numbers within each dimension are randomly permutated. Therefore, to meet the space-filling property, we can simultaneously do these two kinds of random treatment for Maxiter times and get the best two samples conforming the Maximin criterion as the final samples

$$\begin{bmatrix} 0.9741 & 0.8067 & 0.7940 & 0.8658 & 0.1833 \\ 0.2525 & 0.1708 & 0.1659 & 0.1377 & 0.7149 \end{bmatrix}.$$

The total set of samples, which meets the projective and space-filling properties of LHD is then obtained as follows:

$$\begin{bmatrix} 0.4482 & 0.3107 & 0.0979 & 0.3187 & 0.0277 \\ 0.2082 & 0.6744 & 0.5410 & 0.9456 & 0.5343 \\ 0.7267 & 0.9846 & 0.9063 & 0.0850 & 0.2718 \\ 0.8399 & 0.0571 & 0.2664 & 0.7276 & 0.9187 \\ 0.0372 & 0.5325 & 0.7230 & 0.4059 & 0.7774 \\ 0.6167 & 0.4780 & 0.4875 & 0.5089 & 0.4051 \\ 0.9741 & 0.8067 & 0.7940 & 0.8658 & 0.1833 \\ 0.2525 & 0.1708 & 0.1659 & 0.1377 & 0.7149 \end{bmatrix}.$$

5 | CONCLUSIONS

In this paper, a sequential method is proposed for the construction of LHD. The proposed method has the following advantages:

1. Unlike the existing sequential strategies that determine the sampling areas according to a specified metamodel, the proposed method does not depend on any specified metamodel. The proposed method is metamodel independent.
2. The existing sequential sampling strategies only guarantee the projective property locally. They are LHDs in a subspace but not LHDs in the entire design space. The proposed method can guarantee the projective property in the entire design space by utilizing a search algorithm.
3. The existing sequential LHD techniques are time inefficient. As dimensionality grows, the local enumeration process in the S-SLE method[19] becomes more and more computationally expensive, which significantly decreases the sampling efficiency of the S-SLE method for high-dimensional applications. The Quasi-LHD method in Xiong et al[15] also requires a time-consuming global optimization process. Different from the existing methods, the search process in this paper is time efficient. It is very easy to obtain the number of design points that we want to add to the previous stage.
4. To guarantee the space-filling property, Algorithm 1 in this paper is designed by utilizing the Maximin criterion. In this algorithm, the sampling process is repeated for many times at each stage and then one set, which has a maximum value in all of these minimum distances between design points is chosen. Therefore, the proposed method not only can ensure the projective property but also can guarantee the space-filling property in the entire design space.

Two low-dimensional examples and one higher-dimensional example are employed to illustrate the sample-generating process. It is shown that the proposed method has good projective and space-filling properties, which are widely required in industrial design areas involving computationally expensive simulations.

ACKNOWLEDGEMENTS

The funding provided for this study by the National Science Foundation of China under grant numbers 71872088, 71401080, 71802110, and 71702072, the National Security Agent under grant number H98230-15-1-0253, the Social Science Foundation of Jiangsu under grant number 17GLB016, the State Scholarship Fund of China under grant number 201508320059, "1311 Talent Fund" of NJUPT, the Science Foundation of Jiangsu under grant numbers BK20170894 and BK20170810, and Social Science Foundation of NJUPT under grant numbers NYS216011, NY218064, and NYJD217006 are gratefully acknowledged.

REFERENCES

1. Johnson ME, Moore LM, Ylvisaker D. Minimax and maximin distance designs. *J Stat Plann Inference*. 1990;26(2):131-148.
2. Jin R, Chen W, Sudjianto A. An efficient algorithm for constructing optimal design of computer experiments. *J Stat Plann Inference*. 2005;134(1):268-287.
3. Yang X, Yang JF, Lin DK, Liu MQ. A new class of nested (nearly) orthogonal latin hypercube designs. *Stat Sinica*. 2016;26:1249-1267.
4. Chen H, Huang H, Lin DK, Liu MQ. Uniform sliced latin hypercube designs. *Applied Stochastic Models in Business and Industry*. 2016;32(5):574-584.
5. Wang L, Yang JF, Lin DK, Liu MQ. Nearly orthogonal latin hypercube designs for many design columns. *Stat Sinica*. 2015;25:1599-1612.
6. Steinberg DM, Lin D. Construction of orthogonal nearly latin hypercubes. *Qual Reliab Eng Int*. 2015;31(8):1397-1406.
7. Yin Y, Lin DK, Liu MQ. Sliced latin hypercube designs via orthogonal arrays. *J Stat Plann Inference*. 2014;149:162-171.
8. Yang J, Liu MQ, Lin DK. Construction of nested orthogonal latin hypercube designs. *Stat Sinica*. 2014;24(1):211-219.
9. Kenny QY, Li W, Sudjianto A. Algorithmic construction of optimal symmetric latin hypercube designs. *J Stat Plann Inference*. 2000;90(1):145-159.
10. Li WW, Wu JC. Columnwise-pairwise algorithms with applications to the construction of supersaturated designs. *Technometrics*. 1997;39(2):171-179.
11. Sacks J., Schiller SB, Welch WJ. Designs for computer experiments. *Technometrics*. 1989;31(1):41-47.
12. Jin R, Chen W, Sudjianto A. On sequential sampling for global metamodeling in engineering design. In: ASME 2002 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference American Society of Mechanical Engineers; 2002:539-548.
13. Kim B, Lee Y, Choi D-H. Construction of the radial basis function based on a sequential sampling approach using cross-validation. *J Mech Sci Technol*. 2009;23(12):3357-3365.
14. Wang GG. Adaptive response surface method using inherited latin hypercube design points. *J Mech Des*. 2003;125(2):210-220.
15. Xiong F, Xiong Y, Chen W, Yang S. Optimizing latin hypercube design for sequential sampling of computer experiments. *Eng Optim*. 2009;41(8):793-810.
16. Crombecq K, Laermans E, Dhaene T. Efficient space-filling and non-collapsing sequential design strategies for simulation-based modeling. *Eur J Operational Res*. 2011;214(3):683-696.
17. Liu H, Xu S, Wang X. Sequential sampling designs based on space reduction. *Eng Optim*. 2015;47(7):867-884.
18. Zhu H, Liu L, Long T, Peng L. A novel algorithm of maximin latin hypercube design using successive local enumeration. *Eng Optim*. 2012;44(5):551-564.
19. Long T, Wu D, Chen X, Guo X, Liu L. A deterministic sequential maximin latin hypercube design method using successive local enumeration for metamodel-based optimization. *Eng Optim*. 2016;48(6):1019-1036.
20. Zhou XJ, Ma YZ, Li XF. Ensemble of surrogates with recursive arithmetic average. *Struct Multi Optim*. 2011;44(5):651-671.
21. Zhou XJ, Ma YZ, Tu YL, Feng Y. Ensemble of surrogates for dual response surface modeling in robust parameter design. *Qual Reliab Eng Int*. 2012;29(2):173-197.
22. Zhou X, Jiang T. Metamodel selection based on stepwise regression. *Struct Multi Optim*. 2016;54(3):641-657.
23. Ouyang L, Zhou D, Park C, Chen J, Tu Y. Ensemble modelling technique for a micro-drilling process based on a two-stage bootstrap. *Eng Optim*. 2018;51(3):1-17.
24. Zhou XJ, Ma YZ. A study on smo algorithm for solving ϵ-svr with non-psd kernels. *Commun Stat-Simul Comput*. 2013;40(10):2175-2196.
25. Zhou XJ, Jiang T. Enhancing least square support vector regression with gradient information. *Neural Process Lett*. 2016;43(1):65-83.
26. Zhou X, Jiang T. An effective way to integrate ε-support vector regression with gradients. *Expert Syst Appl*. 2018;99:126-140.

AUTHOR BIOGRAPHIES

XiaoJian Zhou is an associate professor in the School of Management, Nanjing University of Posts and Telecommunications (NJUPT), Nanjing, China. His current research interests include design of experiments, response surface methods, metamodeling, and intelligent quality control.

Dr. Dennis K. J. Lin is a University Distinguished Professor of Supply Chain and Statistics at Penn State University. His research interests include quality engineering, industrial statistics, data mining and response surface. He has published over 150 papers in a wide variety of journals. He currently serves as co-editor for Applied Stochastic Models for Business and Industry, and associate editor for various journals: Technometrics, Statistica Sinica, Journal of Quality Technology, Journal of Data Science, Quality Technology & Quality Management, Journal of Statistics and Its Applications, and Journal of Statistical Theory and Practice. Dr Lin is an elected fellow of ASA and ASQ, an elected member of ISI, a lifetime member of ICSA, a fellow of RSS, and a Chang-Jiang Scholar of China at Renmin University. He is also the recipient of the 2004 Faculty Scholar Medal Award at Penn State University.

XueLong Hu is currently an assistant professor in the School of Management, Nanjing University of Posts and Telecommunications. His main areas of interest are in new statistical quality control technique.

Linhan Ouyang is an assistant professor in the Department of Management Science and Engineering, Nanjing University of Aeronautics and Astronautics, China. He received his BS in Industrial Engineering form Nanchang University, China, and a PhD in Quality Engineering from Nanjing University of Science and Technology, China. His research interests include design of experiments and industrial statistics.

How to cite this article: Zhou X, Lin DKJ, Hu X, Ouyang L. Sequential latin hypercube design with both space-filling and projective properties. *Qual Reliab Engng Int*. 2019;1–11. https://doi.org/10.1002/qre.2485

APPENDIX : PROOF OF THEOREM 1

Proof. The proof is direct. When we use the steps from 3 to 6 in *Algorithm 1* to find the number N_{add}, the number N_{add} is what we want to find for k. Since when $k = n$, the set of design points $(x_{ij})_{(n+k)\times d}$ is definitely an LHD, the number that we find for k is definitely no larger than n (ie, $k \leqslant n$). According to the steps from 3 to 6 in *Algorithm 1*, we know that there is no more than one projection point in each of the $n + k$ intervals for any dimension j ($j = 1, \ldots, d$); that is, any two elements in the set $\left\{ \lfloor (n+k)x_{ij} \rfloor, i = 1, \ldots, n \right\}$ are not equal for any dimension j ($j = 1, \ldots, d$). Therefore, such set of design points $(x_{ij})_{(n+k)\times d}$ is an LHD. □