A deterministic sequential maximin Latin hypercube design method using successive local enumeration for metamodel-based optimization

Teng Long*, Di Wu, Xin Chen, Xiaosong Guo and Li Liu

Key Laboratory of Dynamics and Control of Flight Vehicle, Ministry of Education, School of Aerospace Engineering, Beijing Institute of Technology, Beijing, PR China

(Received 19 July 2015; accepted 3 August 2015)

Space-filling and projective properties of design of computer experiments methods are desired features for metamodeling. To enable the production of high-quality sequential samples, this article presents a novel deterministic sequential maximin Latin hypercube design (LHD) method using successive local enumeration, notated as sequential-successive local enumeration (S-SLE). First, a mesh-mapping algorithm is proposed to map the positions of existing points into the new hyper-chessboard to ensure the projective property. According to the maximin distance criterion, new sequential samples are generated through successive local enumeration iterations to improve the space-filling uniformity. Through a number of comparative studies, several appealing merits of S-SLE are demonstrated: (1) S-SLE outperforms several existing LHD methods in terms of sequential sampling quality; (2) it is flexible and robust enough to produce high-quality multiple-stage sequential samples; and (3) the proposed method can improve the overall performance of sequential metamodel-based optimization algorithms. Thus, S-SLE is a promising sequential LHD method for metamodel-based optimization.

Keywords: Latin hypercube design; design of computer experiments; sequential sampling; successive local enumeration; metamodeling; optimization

1. Introduction

In order to save the computational expense of modern engineering design optimization involving high-fidelity simulations (e.g. finite element analysis and computational fluid dynamics models), metamodels, also called surrogate models, have been widely utilized to replace those expensive black-box functions. In the past two decades, significant development of metamodel-based design optimization (MBDO) has been achieved, including novel MBDO algorithms and industrial applications (Simpson et al. 2004; Wang and Shan 2007; Forrester and Keane 2009; Shan and Wang 2010; Viana et al. 2014). As a key technique in the MBDO area, design of computer experiments greatly influences the global accuracy of metamodels and the overall performance of MBDO algorithms including global exploration capability, convergence speed, efficiency and robustness. It is commonly recognized that high-quality samples should possess favourable space-filling and projective properties (Johnson, Moore, and Ylvisaker 1990; Morris and Mitchell 1995). The space-filling property requires points to distribute in the design space as
evenly as possible, and the projective property requires the samples to be uniformly positioned in each dimension or the lower dimensional projective planes (van Dam, Husslage, and den Hertog 2007). The Latin hypercube design (LHD) method naturally guarantees the perfect projective property. In recent years, lots of one-stage optimal Latin hypercube design (OLHD) methods have been developed to improve the space-filling and projective behaviours, such as the maximin LHD method (Johnson, Moore, and Ylvisaker 1990), optimal symmetrical LHD method (Ye, Li, and Sudjianto 2000), enhanced stochastic evolutionary algorithm (ESEA)-based OLHD method (Jin, Chen, and Sudjianto 2005), translational propagation algorithm-based fast optimal Latin hypercube design method (TPLHD) (Viana, Venter, and Balabanov, 2010) and a maximin LHD method using successive local enumeration (SLE) (Zhu et al. 2012).

However, to further reduce computational cost, it is preferred to employ sequential sampling methods to gradually increase samples for updating metamodels. As pointed out by Jin, Chen, and Sudjianto (2002), sequential sampling methods are developed for two major purposes: accurate global metamodelling and efficient metamodel-based optimization. For this first purpose, sequential samples are desired to be placed at the locations where the present metamodel shows poor fitting quality. For instance, Sacks et al. (1989) proposed a sequential sampling algorithm based on the integrated mean squared error (IMSE) criterion to update the kriging metamodel. Jin, Chen, and Sudjianto (2002) proposed two sequential sampling approaches, one based on maximin scaled distance and the other on cross-validation. Kim, Lee, and Choi (2009) used the maximum modified cross-validation error criterion as the objective to generate new points. Assisted by a bootstrapping technique, van Beers and Kleijnen (2008) generated a batch of new samples by repeatedly maximizing the metamodel’s uncertainty. In metamodel-based optimization, sequential sampling approaches are used to assign more samples towards the true global optimum. For that purpose, two kinds of sequential sampling method have been frequently used in MBDO algorithms. Without considering space-filling and projective properties, the first kind of sequential sampling method directly produces new samples in terms of certain infill criteria based on optimality, uncertainty, etc., such as efficient global optimization (EGO) (Jones, Schonlau, and Welch 1998) and mode-pursuing sampling (MPS) algorithm (Wang, Shan, and Wang 2004). On the other hand, the move-limit-based MBDO algorithms increase new samples inside regions of interest where the true global optimum probably exists. So far, different techniques have been developed to identify the regions of interest, such as the trust region-based metamodel management framework (Alexandrov et al. 1998), cutting-plane approach used in the adaptive response surface method (Wang, Dong, and Aitchison 2001), fuzzy clustering-based space-reduction methods (Wang and Simpson 2004; Li et al. 2013) and the significant design (or sampling) space method (Long et al. 2012; Peng et al. 2014). Although one-stage OLHD methods can be directly used for sequential sampling in the regions of interest, this may cause several problems, explained as follows. Because regions of interest are continuously updated during the optimization, certain existing samples probably locate in the current region of interest. However, the one-stage OLHD method cannot ensure the space-filling uniformity between the new samples and existing ones, which may restrict the exploration performance. More importantly, if one-stage OLHD methods produce new samples very close to the existing ones, this is very likely to cause matrix singularity of refitting metamodels, as pointed out by Jones, Schonlau, and Welch (1998). Owing to the aforementioned limitations of one-stage OLHD methods, effective sequential maximin LHD methods are desired to improve the performance of sequential MBDO algorithms. However, to the authors’ knowledge, few studies have reported on sequential maximum LHD methods. Jin, Chen, and Sudjianto (2002) used a maximin scaled distance criterion to select samples from candidate points. Wang (2003) developed an inheriting LHD method to increase new LHD points in reduced design spaces considering the effects of existing points. Although the inheriting LHD method improved the projective uniformity, the space-filling property was not addressed. Xiong et al. (2009) proposed a Quasi-LHD method to achieve good
space-filling and projective behaviours through a genetic algorithm (GA) optimization process. But Quasi-LHD is not as efficient since a global optimization process is required. Crombecq, Laermans, and Dhaene (2011) used a Monte Carlo approach to develop the mc-intersite-proj-th (MIPT) method for sequential sampling, which a employed space-reduction strategy to ensure the projective property of samples. Liu, Xu, and Wang (2015) proposed a sequential maximin LHD approach using a Monte Carlo method and space-reduction strategy (i.e. MCSR), and then a local boundary search algorithm was used to further improve the space-filling property. The Monte Carlo approach introduced randomness into the MIPT method and its variants. Hence, new sequential maximum LHD methods are still expected to upgrade the performance of move-limit-based MBDO algorithms.

In this article, a novel sequential maximum LHD method using successive local enumeration, named sequential-successive local enumeration (S-SLE), is proposed to sequentially produce samples in regions of interest considering the space-filling and projective properties. S-SLE is an extension of a previous one-stage SLE method whose satisfactory performance has been demonstrated, especially for low- and intermediate-dimensional problems (Zhu et al. 2012). A practical mesh-mapping algorithm is first proposed to map the existing points previously locating in the region of interest into the evenly meshed hyper-chessboard. Then, using the local enumeration mechanism, new samples are determined by maximizing the minimal distance between the newly added and existing points. Since no global optimization process is required, the developed S-SLE is expected to efficiently generate high-quality sequential samples.

The rest of the article is organized as follows. Section 2 details the proposed S-SLE method. S-SLE is then compared with several one-stage OLHD methods and sequential LHD methods in Section 3. In Section 4, further comparative studies are presented to show the appealing merits of S-SLE for metamodel-based optimization. Finally, conclusions and directions for future work are given in Section 5.

2. Description of the sequential-successive local enumeration method

To ensure space-filling and projective properties, the proposed S-SLE method consists of two subroutines, namely, the mesh-mapping algorithm to map existing points into the evenly meshed hyper-chessboard, and the sequential maximin sampling algorithm using successive local enumeration based on the maximin distance criterion. The two algorithms are detailed as follows.

2.1. Mesh-mapping algorithm

The original one-stage SLE method gradually positions the samples in a hyper-chessboard to ensure the projective property in each dimension, and a successive local enumeration process is then executed to improve the space-filling property. For two-dimensional (2D) problems, the hyper-chessboard is a square; for three-dimensional (3D) problems, it becomes a cube. For \( n \)-dimensional problems \( (n > 3) \), it can be regarded as a hypercube. According to the SLE algorithm, the samples need to be placed in the available grids in a uniformly meshed hyper-chessboard based on the number of samples \( (m) \). Each grid is one element of the meshed hyper-chessboard, which has the same geometric topology as the hyper-chessboard. Since the existing samples may locate at arbitrary positions, the authors derived a mesh-mapping algorithm to identify the grids that have been occupied by existing points to ensure the projective property of sequential sampling. Assume that the number of existing points is \( m_1 \); the number of sequential samples is \( m_2 \); and the existing points set is represented by \( P^{ex} = \{P^{ex}_1, P^{ex}_2, \ldots, P^{ex}_{m_1}\} \), where
\( \bar{P}_{ex} = (\bar{p}_k^{(1)}, \bar{p}_k^{(2)}, \ldots, \bar{p}_k^{(n)}) \) and \( \bar{p}_k^{(j)} \) stands for the \( j \)th coordinate of the \( k \)th existing point. The procedure of the \( n \)-dimensional mesh-mapping algorithm is explained below.

Step 1. (Normalization): The current region of interest, also called the design space for simplification, and the existing points are first normalized according to Equation (1). In Equation (1), \( \Theta \) indicates the original coordinates, \( \Theta_{\max} \) and \( \Theta_{\min} \) are the upper and lower bounds of the current design space, and \( \bar{\Theta} \) is the normalized coordinates lying in the interval [0,1]. The normalized design space becomes a unit hypercube.

\[
\bar{\Theta} = \frac{\Theta - \Theta_{\min}}{\Theta_{\max} - \Theta_{\min}}
\]

Step 2. (Generate evenly meshed hyper-chessboard): Since the sequential sampling process of the proposed method is executed cell by cell, the cell number should be equal to the number of total samples including existing points and sequential samples, i.e. \( m = m_1 + m_2 \). Thus, the first dimension of the design space is divided into \( m = m_1 + m_2 \) parts to form \( m \) cells. Then, each of the other dimensions is divided into \( m_0 = m + c \) parts to construct denser meshes for achieving better space-filling performance, where \( c \) is a positive integer called the mesh density parameter. The influence of parameter \( c \) is analysed in Section 2.3 to find the recommended value of \( c \). In the first dimension, each part has the equivalent size of \( 1/m \), and in the other dimensions, the size is \( 1/m_0 \). In terms of \( m, m_0 \) and \( n \), the design space can then be converted into an evenly meshed hyper-chessboard with \( m \times m_0^{n-1} \) grids. The position of an arbitrary grid in the hyper-chessboard can be expressed by a set of integers \((I_1, I_2, \ldots, I_n)\), where \( I_1 \in \{1,2,\ldots,m\} \) and \( I_i \in \{1,2,\ldots,m_0\}, i=2,\ldots,n. \)

Step 3. (Map the coordinate of one existing point): The normalized coordinates of the \( k \)th existing point can be expressed as \( \bar{P}_{ex} = (\bar{p}_k^{(1)}, \bar{p}_k^{(2)}, \ldots, \bar{p}_k^{(n)}) \). To map the first dimension of the \( k \)th existing point \( \bar{p}_k^{(1)} \), first set \( j = 1 \), and then compare \( \bar{p}_k^{(1)} \) with \( j/m \). If \( \bar{p}_k^{(1)} < j/m \), the mapped value \( \bar{p}_k^{(1)} \) is set to \( j \). Otherwise, set \( j = j + 1 \) and continue until \( \bar{p}_k^{(1)} < j/m \). For the other dimensions of the \( k \)th existing point, compare \( \bar{p}_k^{(j)} \) with \( j/m_0 \) to determine the mapped value \( \bar{p}_k^{(j)} \) through the same process as above. After that, the mapped coordinates of the existing points in the hyper-chessboard can be expressed by an integer set \( \bar{P}_{ex} = (\bar{p}_k^{(1)}, \bar{p}_k^{(2)}, \ldots, \bar{p}_k^{(n)}) \), which indicates the grid already occupied by the \( k \)th existing sample.

Step 4. (Map all the existing points and output): Step 3 is repeated to map all the existing points in the same manner. The mapped coordinates of all the existing points, as expressed by an integer matrix \( \bar{P}_{ex} = (\bar{P}_{ex}^1, \bar{P}_{ex}^2, \ldots, \bar{P}_{ex}^m) \), and the normalized coordinates of existing points \( \bar{P}_{ex} \) are output for the subsequent sequential maximin sampling process.

Output: the locations of the existing points in the \( m \times m_0^{n-1} \) hyper-chessboard \( \bar{P}_{ex} = (\bar{P}_{ex}^1, \bar{P}_{ex}^2, \ldots, \bar{P}_{ex}^m) \), where \( \bar{P}_{ex} = (\bar{p}_k^{(1)}, \bar{p}_k^{(2)}, \ldots, \bar{p}_k^{(n)}) \), and \( \bar{p}_k^{(j)} \) is an integer between 1 and \( m_0 \); the normalized coordinates of existing points \( \bar{P}_{ex} = (\bar{P}_{ex}^1, \bar{P}_{ex}^2, \ldots, \bar{P}_{ex}^m) \), where \( \bar{P}_{ex} = (\bar{p}_k^{(1)}, \bar{p}_k^{(2)}, \ldots, \bar{p}_k^{(n)}) \).

In order to illustrate the mesh-mapping algorithm clearly, a 2D example is presented, where the hyper-chessboard degenerates to a square. Assume that four points already exist in the design space (i.e. \( m_1 = 4 \)), as shown in Figure 1(a), and another four sequential samples need to be generated (i.e. \( m_2 = 4 \)). The number of total samples is thus equal to eight (i.e. \( m = 8 \)). By setting \( c = 4 \), the first dimension is divided into eight parts, while the second dimension is divided into 12 parts (i.e. \( m_0 = 12 \)). For the first existing point (from the left), the first dimension \( \bar{p}_1^{(1)} \) satisfies the inequality \( \bar{p}_1^{(1)} = 0.054 < 1/8 \), so the mapped value is set to 1 (i.e. \( \bar{p}_1^{(1)} = 1 \)). Similarly, the mapped value of the second dimension for the first point is 4 (i.e. \( \bar{p}_1^{(2)} = 4 \)). Thus, the mapped position of the first point in the grid is (1,4) in the plane chessboard, as shown in
Figure 1(b). In the same way, the mapped coordinates of all the existing points can be determined, i.e., \( \tilde{P}_{\text{ex}} = (\tilde{P}_{\text{ex}}^1, \tilde{P}_{\text{ex}}^2, \ldots, \tilde{P}_{\text{ex}}^{m_1}) \) = \{(1, 4), (3, 11), (5, 8), (7, 1)\}, as depicted in Figure 1(b).

2.2. Sequential maximin sampling algorithm

Once the mapped coordinates of the existing points have been obtained, a sequential maximin sampling algorithm using successive local enumeration inherited from one-stage SLE method can be used to generate the desired sequential samples. In the \( m \times m^{n-1} \) unit hyper-chessboard, the new sequential points are determined cell by cell, and only one point can be designated in each cell that consists of \( (m_c)^{n-1} \) grids. Algorithm 1 presents the procedure of the sequential maximin sampling algorithm for \( n \)-dimensional problems.

Step 1. (Update the available grids for the selected cell, lines 2–3): According to the principle of LHD, every dimension of newly added samples has to differ from those occupied by existing points. Similar to SLE, the S-SLE method generates sequential samples cell by cell. All the cells are numbered with integers from 1 to \( m \) according to their distance from the origin. All cells are checked one by one until the first cell without any existing points is found, which is selected for generating the \( j \)th new sample. The first dimension of the \( j \)th sequential sample \( \tilde{P}_{\text{seq}}^{(1)} \) is set to the selected cell number. In the evenly meshed hyper-chessboard, candidate positions in the remaining \( n-1 \) dimensions are represented by integers from 1 to \( m_c \), also called identifiers.

To obtain available grids, the identifiers that are not occupied by existing samples are stored in an \( (m_2 + c + 1 - j) \times (n-1) \) matrix \( \tilde{P}_{\text{ava}} \). For the \( j \)th new sample, each column of \( \tilde{P}_{\text{ava}} \) stores \( m_2 + c + 1 - j \) candidate identifiers in each of the remaining \( n - 1 \) dimensions. Since the first dimension of the \( j \)th new sample has been already determined, \( n - 1 \) rows of matrix \( \tilde{P}_{\text{ava}} \) are used to record the candidate information for the remaining dimensions. Columnwise permutation on matrix \( \tilde{P}_{\text{ava}} \) is performed to form the other \( n - 1 \) dimensions of the available grids. The number of available grids for the \( j \)th sequential point is \( (m_2 + c + 1 - j)^{n-1} \). \( \tilde{P}_{\text{ava}} \) is a matrix whose \( l \)th row is \( \tilde{P}_{\text{ava}}^l \), i.e., the \( l \)th available grid, \( l = 1, 2, \ldots, (m_2 + c + 1 - j)^{n-1} \).

\[
\tilde{p}_{\text{ava}(i)} = \frac{2(\tilde{p}_{\text{ava}(i)} - 1) + 1}{2m}, \quad i = 1, 2, \ldots, n
\]  (2)

Step 2. (Determine the \( j \)th potential grid set \( \tilde{P}_{\text{po}}^j \), lines 4–15): Since the centers of available grids are used to compute the distances, the normalized coordinate of the \( l \)th available grid can be calculated in Equation (2), where \( \tilde{P}_{\text{ava}}^l = (\tilde{p}_{\text{ava}(1)}, \tilde{p}_{\text{ava}(2)}, \ldots, \tilde{p}_{\text{ava}(n)}) \) is the integer coordinate. The coordinates of all the available grids can be normalized to obtain matrix \( \tilde{P}_{\text{ava}}^j \).
Algorithm 1 Sequential maximin sampling using successive local enumeration

Input: number of existing points $m_1$; mapped coordinates of existing points $\hat{P}^e = \{\hat{P}_1^e, \hat{P}_2^e, \ldots, \hat{P}_{m_1}^e\}$; normalized coordinates of existing points $\bar{P}^e = \{\bar{P}_1^e, \bar{P}_2^e, \ldots, \bar{P}_{m_1}^e\}$; number of sequential samples $m_2$; mesh density parameter $c$; number of candidate points $s$; minimum allowable projective distance $d_{allow}$

Output: normalized newly added sequential samples set $\hat{P}^{\text{seq}} = \{\hat{P}_1^{\text{seq}}, \hat{P}_2^{\text{seq}}, \ldots, \hat{P}_{m_2}^{\text{seq}}\}$; normalized entire samples set $\bar{P} = \bar{P}^e \cup \hat{P}^{\text{seq}} = \{\bar{P}_1^e, \bar{P}_2^e, \ldots, \bar{P}_{m_1}^e, \bar{P}_1^{\text{seq}}, \ldots, \bar{P}_{m_2}^{\text{seq}}\}$

1: for $j = 1$ to $m_2$
2: Select a cell;
3: Identify all the available grids $\hat{P}^{ava}$ in the selected cell;
4: if $(m_2 + 1 - j)^{n-1} \leq n_p$ then
5: $\hat{P}^{po} = \hat{P}^{ava}$;
6: $h \leftarrow (m_2 + 1 - j)^{n-1}$;
7: else then
8: normalize the coordinates of available grids $\hat{P}^{ava}$ to get $\bar{P}^{ava}$;
9: for $l = 1$ to $(m_2 + 1 - j)^{n-1}$
10: calculate $d_{\min,l}$ between $\bar{P}^{ava}_l$ and $\bar{P}^e$;
11: end
12: sort $d_{\min,l}$ in a descending order;
13: determine the $\bar{P}^{po}$ from selecting $n_p$ potential grids;
14: $h \leftarrow n_p$;
15: end
16: $d_{\max,m} \leftarrow 0$;
17: for $t = 1$ to $h$
18: $d_{\max,m,t} \leftarrow 0$;
19: Generate $s$ candidate points $\tilde{\bar{P}}^{\text{can}(q)}_t (q = 1, 2, \ldots, s)$ in $t$th potential grid using TPLHD;
20: for $q = 1$ to $s$
21: calculate $d^p_{\min,q}$ between $\tilde{\bar{P}}^{\text{can}(q)}_t$ and $\bar{P}^e$;
22: if $d^p_{\min,q} < d_{allow}$ then
23: $d_{\min,q} \leftarrow 0$;
24: else then
25: calculate $d_{\min,q}$ between $\tilde{\bar{P}}^{\text{can}(q)}_t$ and $\bar{P}^e$;
26: end
27: if $d_{\min,q} > d_{\max,m,t}$ then
28: $d_{\max,m,t} \leftarrow d_{\min,q}$; $\tilde{\bar{P}}^{po}_t \leftarrow \tilde{\bar{P}}^{\text{can}(q)}_t$;
29: end
30: end
31: if $d_{\max,m,t} > d_{\max,m}$ then
32: $d_{\max,m} \leftarrow d_{\max,m,t}$; $\tilde{\bar{P}}^{\text{seq}}_j \leftarrow \tilde{\bar{P}}^{po}_t$; $\tilde{\bar{P}}^{\text{seq}}_j \leftarrow \tilde{\bar{P}}^{\text{can}(q)}_t$;
33: end
34: end
35: $\bar{P}^e \leftarrow \bar{P}^e \cup \tilde{\bar{P}}^{\text{seq}}_j$; $\tilde{\bar{P}}^s \leftarrow \tilde{\bar{P}}^{\text{seq}}_j \cup \tilde{\bar{P}}^{\text{seq}}_j$; $\bar{P}^e \leftarrow \bar{P}^e \cup \tilde{\bar{P}}^{\text{seq}}_j$;
36: end
37: Return $\bar{P}^{\text{seq}}, \bar{P}$
Distances between the \( l \)th available grid \( \bar{P}^{\text{ava}} \) and all the existing points \( \bar{P}^{\text{ex}}(k = 1, 2, \ldots, m_1 + j − 1) \) are then calculated to identify the minimal distance (i.e. \( d_{\text{min},j} \)), which is defined as the eigenvalue of the \( l \)th available grid. Next, those eigenvalues are sorted in descending order and the available grids with the top \( n_p \) largest eigenvalues in the selected cell are considered as the potential grids \( \bar{P}^{\text{po}}(t = 1, 2, \ldots, n_p) \), which are stored in the set of potential grids \( \bar{P}^{\text{po}} \). The desired point with the best projective and space-filling properties probably locates in the design space represented by \( \bar{P}^{\text{po}} \). In this study, the number of potential grids is set to five. If the number of available grids is less than five, all of them are chosen as potential grids.

Step 3. (Determine the potential point \( \bar{P}^{\text{po}} \) in the \( r \)th potential grid, lines 18–30): Since it cannot be guaranteed that the centre point of each potential grid has the best space-filling and projective properties, a subsampling process is executed in each potential grid \( \bar{P}^{\text{po}} \) to pursue better space-filling performance. TPLHD (Viana, Venter, and Balabanov 2010), a deterministic one-stage maximin LHD method, is employed to generate \( s \) candidate points \( \bar{P}^{\text{can}(q)} \) \((q = 1, 2, \ldots, s)\) in the \( t \)th potential grid. The minimal allowable projective distance, i.e. \( d_{\text{max},t}^{\text{allow}} = 0.5/(m − 1) \), is employed to remove the points with poor projective property.

If the minimum projective distance between the candidate point \( \bar{P}^{\text{can}(q)} \) and any existing point is less than \( d_{\text{max},t}^{\text{allow}} \), the minimal distance (i.e. \( d_{\text{min},q} \)) is set to zero, which indicates a non-promising point. Otherwise, the actual distances between the candidate point and existing points are calculated. The candidate point with the maximal \( d_{\text{min},q} \) (i.e. \( d_{\text{max},t} \)) is defined as the potential point \( \bar{P}^{\text{po}} \) in the \( r \)th potential grid.

Step 4. (Determine the \( j \)th sequential sample \( \bar{P}^{\text{seq}} \), lines 16–34): Step 3 is repeated to identify all the potential points. Next, the potential point with the maximal \( d_{\text{max},t} \) (i.e. \( d_{\text{max},t} \)) is picked as the \( j \)th sequential sample \( \bar{P}^{\text{seq}} \). The potential grid expressed by integers that contains the \( j \)th sequential sample is added to the existing samples set \( \bar{P}^{\text{ex}} \) to update the available grids for subsequent iterations.

Step 5. (Determine all the \( m_2 \) sequential samples, lines 1–37): Steps 1–4 are repeated to generate all the \( m_2 \) sequential samples, which are also saved in the sequential samples set \( \bar{P}^{\text{seq}} \) and entire samples set \( \bar{P} \). Since the output samples in \( \bar{P}^{\text{seq}} \) and \( \bar{P} \) are still normalized values locating inside the unity hypercube, users can perform the inverse operation of Equation (1) to obtain the actual coordinates.

The same 2D sequential sampling example in the previous section is used to illustrate the procedure of Algorithm 1, as shown in Figure 2. Note that the mesh-mapping of the four existing points has been accomplished, as shown Figure 1(b). For this 2D problem, a cell is defined as a column consisting of 12 squares (i.e. grids). After eliminating all the columns occupied by the existing points to ensure the projective property, column 2 is the first available cell where no existing point locates, so the first dimension for the first sequential sample is set to integer 2. In column 2, the available identifiers for the second dimension include 2, 3, 5, 6, 7, 9, 10 and 12, which means eight candidate grids, as marked by shaded squares in Figure 2 (a), need to be inspected to determine the first sequential sample. According to Algorithm 1, matrix \( \hat{P}' \) can be expressed as \( \hat{P}' = (2, 3, 5, 6, 7, 9, 10, 12)^T \). By combining integer 2 with every row of \( \hat{P}' \), the available grids can be expressed by \( \bar{P}^{\text{ava}} \):

\[
\bar{P}^{\text{ava}} = \begin{bmatrix}
2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\
2 & 3 & 5 & 6 & 7 & 9 & 10 & 12
\end{bmatrix}^T
\]  

In terms of Equation (2), normalization of \( \bar{P}^{\text{ava}} \) is performed to obtain \( \bar{P}^{\text{ava}} \) as

\[
\bar{P}^{\text{ava}} = \begin{bmatrix}
0.188 & 0.188 & 0.188 & 0.188 & 0.188 & 0.188 & 0.188 \\
0.125 & 0.208 & 0.375 & 0.458 & 0.542 & 0.708 & 0.792 & 0.958
\end{bmatrix}^T
\]
Figure 2. Sequential maximin sampling process on a two-dimensional problem \((m_1 = 4, m_2 = 4, c = 4, s = 5)\). (a) Initial samples; (b) candidate points; (c) potential points; (d) first sample; (e) final sample.

The eigenvalues of the available grids are 0.193, 0.144, 0.173, 0.236, 0.308, 0.198, 0.137 and 0.139, respectively. Through sorting those minimal distances, the potential grids set \(\tilde{P}_p\), consisting of the promising grids with the top five largest eigenvalues, is selected, as shown in Equation (5). The potential grids are marked by red-shaded squares with 45° bars (\(/\)) in Figure 2(a), and the black-shaded ones with –45° bars (\(/\)) are discarded non-promising grids.

\[
\tilde{P}_p = \begin{bmatrix}
2 & 2 & 2 & 2 & 2 \\
7 & 6 & 9 & 2 & 5 \\
\end{bmatrix}^T
\]  

(5)

Next, a subsampling process is applied for all the potential grids to determine the potential points \(\bar{P}_p\). Five TPLHD candidate points generated in the first potential grid \(\tilde{P}_p^1 = (2, 7)\) can be expressed in Equation (6), as marked by the four blue circles and one red diamond in Figure 2(b).

\[
\bar{P}_1^{\text{can}} = \begin{bmatrix}
0.188 & 0.250 & 0.125 & 0.219 & 0.156 \\
0.500 & 0.521 & 0.542 & 0.562 & 0.583 \\
\end{bmatrix}^T
\]  

(6)

The minimal projective distance between each candidate point and all existing samples is calculated to discard the points with undesired projective performance. In this example, since the minimal projective distance values of candidates 2, 3, 4 and 5 are less than the minimal allowable distance \(d_{\text{allow}}\) (i.e. 0.071), the eigenvalues of those candidates (i.e. \(d_{\min,2}, d_{\min,3}, d_{\min,4}\) and \(d_{\min,5}\)) are set to zero. Candidate 1, with the largest the eigenvalue (i.e. \(d_{\min,1} = 0.271\)), is determined as the potential point in potential grid \(\tilde{P}_p^1\) [i.e. \(\bar{P}_1 = (0.188, 0.500), d_{\min,1} = 0.271\)], as marked by the red diamond in Figure 2(b). In the same way, the potential points for other potential grids can also be produced, i.e. \(\bar{P}_2 = (0.219, 0.479)\) \(d_{\min,2} = 0.270\), \(\bar{P}_3 = (0.156, 0.750)\) \(d_{\min,3} = 0.187\), \(\bar{P}_4 = (0.219, 0.146)\) \(d_{\min,4} = 0.203\) and \(\bar{P}_5 = (0.219, 0.396)\) \(d_{\min,5} = 0.211\), as shown in Figure 2(c). \(\bar{P}_1 = (0.188, 0.500)\), with the largest eigenvalue, is
then determined as the first sequential point $\bar{P}_{1}^{\text{seq}}$, as indicated by the red star in Figure 2(d), and the grid $\bar{P}_{1}^{\text{po}} = (2, 7)$ containing $\bar{P}_{1}^{\text{po}}$ is added into the existing samples set $\bar{P}_{1}^{\text{ex}}$. Similarly, the remaining three sequential points can be generated, and Figure 2(e) shows the initial points and newly added sequential samples. As can be seen in Figure 2(e), the sequential samples do not become overcrowded with the previous points and all the samples have favourable space-filling and projective properties.

2.3. Influence analysis of parameter $c$

The mesh density parameter $c$ is introduced to control how many grids should be generated in the hyper-chessboard. It is expected that denser meshes will lead to better space-filling performance but higher computational cost. Thus, the appropriate value of $c$ is beneficial to balancing the effectiveness and efficiency of the S-SLE method. To conduct an analysis of the influence of parameter $c$ on space-filling performance and sampling efficiency, the S-SLE method with various values of $c$ was used to generate sequential samples on four problems. Ten initial samples from the MATLAB \textit{lhsdesign} function were generated for all the problems. Note that for each problem, S-SLE sampling processes using difference $c$ values were performed based on the identical initial samples set. In each sampling problem, parameter $c$ was gradually increased from 0 to 10 with a unit increment. Each experiment was repeated 30 times to record the average values of $d_{\text{min}}$ (indicated by circles–line) and sampling time (second) (indicated by squares–line), as shown in Figure 3.

Figure 3 shows that for all the cases the increasing $c$ value leads to larger $d_{\text{min}}$ and sampling time, which is consistent with expectations. For smaller values of $c$, parameter $c$ has a significant positive influence on both $d_{\text{min}}$ and CPU time. However, when $c$ reaches a certain value (e.g.
4 or 5), the improvement in the space-filling property slows down as parameter $c$ continuously increases, but the sampling expense still has a significant increasing trend. To achieve a good trade-off between sampling quality and efficiency, the value of parameter $c$ should range from 4 to 8.

2.4. Properties of the proposed sequential-successive local enumeration method

Since the S-SLE method is derived from the previous one-stage SLE method, all the merits of SLE are fully inherited, and the limitations of SLE have also been solved or at least alleviated. First, by using the mesh-mapping and sequential maximin sampling algorithms, S-SLE is capable of generating sequential samples with good projective and space-filling properties. Secondly, the first sequential point is identified through local enumeration instead of random selection, which eliminates the effect of the first random point on the sampling quality. Thirdly, instead of simply designating sequential samples at the grid centres, a subsampling process using the deterministic TPLHD method is performed in potential grids to find higher quality potential points. Moreover, S-SLE is a deterministic sequential sampling method, which always produces a constant sequential samples set for the given existing samples and number of sequential samples. Finally, because of its favourable flexibility, S-SLE needs to produce only a small amount of new samples in the region of interest at each iteration even for high-dimensional problems, and then gradually increase the samples via a multiple-stage sampling pattern.

Figure 4(a) and (b) illustrate the 20 samples generated from a two-stage sampling process using SLE and S-SLE, respectively. For both methods, the same 10 existing samples were generated by MATLAB lhsdesign function using the 'maximin' criterion within 200 iterations. Note that the same setting of lhsdesign is also used for the following sections. It is clear that the results for S-SLE have better space-filling and projective properties than those for SLE.

3. Comparative studies on sampling performance

In this section, the S-SLE method was compared with several one-stage OLHD methods and sequential maximin LHD methods through a number of sequential sampling experiments.

3.1. Comparison with one-stage optimal Latin hypercube design methods

A few experiments with different dimensionalities (i.e. 2D, 4D and 6D) were carried out for comparison. For each experiment, it was assumed that 20 points already existed in the design
space and 20 sequential samples needed to be generated. The \textit{lhsdesign} function was used to produce 10 different sets of 20 existing samples for all the experiments. Based on the same initial samples, sequential points were produced using \textit{lhsdesign}, ESEA (Jin, Chen, and Sudjianto 2005), TPLHD (Viana, Venter, and Balabanov 2010), SLE (Zhu et al. 2012) and the proposed S-SLE. In this work, the online MATLAB implementations of TPLHD and ESEA developed by Viana (2011) were used. In the S-SLE method, mesh density parameter \(c\) is set to 5 and the number of TPLHD candidate points \(s\) is set to 500, which were also used for the following experiments. To reduce random variations, each LHD method was consecutively executed 10 times on all the experiments based on the same initial \textit{lhsdesign} points. All the experiments in this article were conducted in MATLAB\textsuperscript{®} release 2012a, installed on a desktop computer equipped with an Intel Core 2 Duo 2.93 GHz CPU and 2 G RAM. After normalizing the samples to \((0 \sim 1)\), a bar plot of mean values of the maximin distance criterion (i.e. \(d_{\text{min}}\)) was produced, as shown in Figure 5.

As can be seen from Figure 5, the proposed S-SLE outperforms all the other one-stage OLHD methods in terms of the \(d_{\text{min}}\) space-filling uniformity criterion.

3.2. \textbf{Comparison with sequential maximin Latin hypercube design method}

To illustrate the performance of the proposed method, several existing sequential maximin LHD methods were involved in the comparison, including Quasi-LHD (Xiong et al. 2009), MIPT (Crombecq, Laermans, and Dhaene 2011) and MCSR (Liu et al. 2015).

Several criteria have been developed to evaluate the space-filling uniformity of samples. Besides the \(d_{\text{min}}\) criterion, three other widely used criteria were considered: \(\phi_p\), potential energy \(U\), and centred \(L_2\) discrepancy criterion \(CL_2\) (Jin, Chen, and Sudjianto 2005; Viana, Venter, and Balabanov 2010). In this study, the positive integer \(p\) in the \(\phi_p\) criterion is set to 50. Note that a larger \(d_{\text{min}}\) or smaller values of \(\phi_p\), \(U\) and \(CL_2\) indicate better space-filling properties. Moreover, two projective metrics (i.e. \(\mu_{d_{\text{min}}}\) and \(\mu_{\text{var}}\)) used in Quasi-LHD (Xiong et al. 2009) were also used to compare the projective property. Larger \(\mu_{d_{\text{min}}}\) and smaller \(\mu_{\text{var}}\) indicate better projective properties. The formulae of \(\mu_{d_{\text{min}}}\) and \(\mu_{\text{var}}\) can be found in Xiong et al. (2009).

Comparisons between the S-SLE method and the other three sequential LHD methods were performed for 2D, 3D and 6D problems in terms of the aforementioned criteria. For all the problems, it was assumed that 10 initial points existed and 20 sequential samples needed to be generated using S-SLE, Quasi-LHD, MIPT and MCSR. Based on the same \textit{lhsdesign} initial sample sets, both sequential sampling processes were performed 10 times to record the average values of various criteria, as shown in Table 1. It is worth noting that the S-SLE method generates
Table 1. Summary of comparison results with three sequential maximin Latin hypercube design (LHD) methods.

| n  | Method     | \(d_{\text{min}}\) | \(\phi_p\) | \(U\)   | \(CL_2\) | \(\mu_{d_{\text{min}}}\) | \(\mu_{\text{var}}\) | \(t\) (s) |
|----|------------|---------------------|-----------|---------|---------|-----------------|-----------------|---------|
| 2  | Quasi-LHD  | 0.093               | 10.578    | 3205.1  | 0.057   | 0.015           | 2.2e−4         | 49.442  |
|    | MIPT       | 0.126               | 6.529     | 3210.1  | 0.048   | 0.016           | 1.4e−4         | 7.756   |
|    | MCSR       | 0.144               | 5.816     | 3029.9  | 0.045   | 0.016           | 2.1e−4         | 1.256   |
|    | S-SLE      | 0.145               | 5.708     | 3181.1  | 0.034   | 0.016           | 2.1e−4         | 2.898   |
| 3  | Quasi-LHD  | 0.176               | 4.427     | 1606.6  | 0.057   | 0.014           | 5.8e−4         | 41.624  |
|    | MIPT       | 0.238               | 3.050     | 1344.0  | 0.066   | 0.017           | 1.3e−4         | 11.637  |
|    | MCSR       | 0.285               | 2.762     | 1256.1  | 0.074   | 0.016           | 1.9e−4         | 1.482   |
|    | S-SLE      | 0.288               | 2.643     | 1348.2  | 0.064   | 0.017           | 2.4e−4         | 3.330   |
| 6  | Quasi-LHD  | 0.320               | 2.169     | 342.4   | 0.582   | 0.013           | 4.2e−4         | 43.968  |
|    | MIPT       | 0.278               | 1.816     | 554.8   | 0.195   | 0.004           | 3.6e−4         | 32.589  |
|    | MCSR       | 0.635               | 0.918     | 443.8   | 0.182   | 0.013           | 9.3e−4         | 37.540  |
|    | S-SLE      | 0.575               | 0.994     | 427.0   | 0.267   | 0.014           | 5.4e−4         | 332.325 |

Note: MIPT = mc-intersite-proj-th; MCSR = Monte Carlo method and space reduction strategy; S-SLE = sequential-successive local enumeration.

Figure 6. Plots of two-dimensional sequential sampling results obtained by (a) Quasi-Latin hypercube design (LHD); (b) mc-intersite-proj-th (MIPT); and (c) Monte Carlo method and space reduction strategy (MCSR) methods (\(n = 2, m_1 = 10, m_2 = 10\)). Solid circles indicate existing points and hollow circles indicate sequential points.

As shown in Table 1, Quasi-LHD generally performs worse in terms of the space-filling criteria compared with the MCSR and S-SLE methods. For the MIPT method, its results for 2D and 3D problems are better than Quasi-LHD but a little worse than MCSR and S-SLE; however, the results of the MIPT method for the 6D problem show the poorest quality. The results in Table 1 reveal that MCSR and S-SLE generally outperform the other two competitors. In 2D and 3D problems, S-SLE shows the best space-filling performance; however, for the 6D problem, the MCSR is superior to S-SLE. Moreover, comparing the values of \(\mu_{d_{\text{min}}}\) and \(\mu_{\text{var}}\), the projective uniformity of S-SLE samples is better than that of the other three sequential LHD methods. In terms of sequential sampling quality, the comparison results thus demonstrate the effectiveness of the proposed S-SLE method for sequential sampling: it obviously outperforms Quasi-LHD and MIPT methods and shows comparable performance with MCSR. Figure 6 shows the 2D sequential sampling results of Quasi-LHD, MIPT and MCSR.

In addition, the sequential sampling efficiency of the four methods was compared; this is measured by the CPU time consumed to generate sequential samples, as listed in the last column of Table 1. For the low-dimensional problems, sequential sampling of S-SLE and MCSR is remarkably faster than for Quasi-LHD. However, as the dimensionality increases, the increased
Figure 7. Box plots of minimal distance of samples obtained by Quasi-Latin hypercube design (LHD), mc-intersite-proj-th (MIPT), Monte Carlo method and space reduction strategy (MCSR), and sequential-successive local enumeration (S-SLE) \(m_1 = 20, m_2 = 20\). (a) Two-dimensional experiments; (b) four-dimensional experiments.

Like the deterministic S-SLE, for the same existing samples, different runs of the other three sequential LHD methods could produce varying results with fluctuating quality. The randomness of the Quasi-LHD method is caused by the GA optimization process, and the randomness of MCSR and MIPT is caused by the Monte Carlo approach. Taking 2D and 4D sequential sampling experiments for example, based on the 20 identical initial samples, the four methods were each performed 10 times to produce 20 new samples. Figure 7 shows the box plots of \(d_{\text{min}}\) values of sampling results obtained by S-SLE, Quasi-LHD, MIPT and MCSR. As one can see from Figure 7, S-SLE does not produce any variation in the space-filling property. In contrast, the results of the other three methods show different degrees of variation for both 2D and 4D experiments, and some poor results of Quasi-LHD and MIPT are obviously inferior to those of S-SLE samples. Note that although the variations in MCSR are much smaller, this method still shows a large likelihood of producing poorer results compared with the constant results of S-SLE. Moreover, \(d_{\text{min}}\) values of S-SLE samples are larger than the median values of the other three methods’ results. In sequential sampling applications, the uncertainty of Quasi-LHD, MIPT and MCSR may produce poor results, while S-SLE guarantees the generation of sequential samples with favourable quality.

It is worth noting that the original Quasi-LHD, MIPT and MCSR can also be modified to perform sequential sampling in a deterministic fashion. For example, Quasi-LHD can employ a deterministic global search algorithm \(e.g.,\) DIRECT to generate sequential points, and MIPT and MCSR can also use deterministic sampling methods \(e.g.,\) TPLHD to generate candidates. In future work, the authors will compare S-SLE with the deterministic versions of Quasi-LHD, MIPT and MCSR in terms of space-filling and projective performance.

The comparison results with Quasi-LHD, MIPT and MCSR reveal that the proposed S-SLE method can effectively generate sequential samples with favourable space-filling and projective properties. S-SLE is more efficient for low-dimensional problems, but its sampling efficiency for high-dimensional applications still needs to be improved. The deterministic sampling behaviour of S-SLE is another advantage over the other methods. In addition to the two-stage sequential sampling applications presented in previous sections, as a sequential maximin LHD method, S-SLE is flexible and can thus support multiple-stage sequential sampling. Thus, S-SLE can generate high-dimensional samples in multiple stages, which improves its efficiency without greatly decreasing the sampling quality.
4. Benefits of sequential-successive local enumeration to sequential metamodel-based optimization

In this section, the sequential radial basis function (SRBF) method using the improved significant sampling space (ISSS) method (a specific move-limit approach) (Peng et al. 2014) was employed to demonstrate the benefits of the proposed S-SLE to sequential metamodel-based optimization.

During the SRBF optimization, significant sampling space (SSS) with a varying size that is identified according to the current acquired knowledge of the underlying optimization problem moves towards the actual global optimum, and inside each SSS certain existing points may locate at arbitrary positions. More detailed description of SRBF can be found in the literature (Peng et al. 2014). The original SRBF employs the lhsdesign function to generate initial and sequential samples in the renewed SSS.

First, the SRBF using the S-SLE method for sequential sampling is compared with the SRBF with lhsdesign and the SLE method. Then, several versions of SRBF using various sequential LHD methods (including Quasi-LHD, MIPT, MCSR and S-SLE) for sequential sampling were developed for comparison. Several well-known benchmark problems, as formulated in Appendix A (see Supplementary material), were solved using different versions of SRBF. For the 2D problems, the number of initial and sequential samples is 12 and 6, respectively, while for higher dimensional ($n > 2$) problems, the number of initial and sequential samples is 20 and $2n$. The convergence tolerance and constraints violation tolerance are set to 1e-4 and 1e-2, respectively. For all the problems, 10 runs of each SRBF were performed, and the average values of optimal solutions ($f^*$) and number of function evaluations ($Nfe$) are presented in Tables 2 and 3 to compare the overall performance. As can be seen in Table 2, the average optimal solutions of SRBF incorporated with the S-SLE method was closer to the theoretical global optima, especially for the Peak (PK), Rastrigin (RS), Hartman (HN) and internal combustion engine design (ICED) (Wagner and Papalambors 1991) problems. Furthermore, for all the problems except for SUR-T1-14 (HD1), SRBF with S-SLE used far fewer $Nfe$ than SRBFs using lhsdesign and SLE. Note that although SRBF with lhsdesign used fewer $Nfe$ to solve the HD1 problem, its solution is inferior to that of S-SLE.

The comparison results of the various SRBFs with different sequential maximin LHD methods are shown in Table 3. SRBF with sequential LHD methods can converge to better solutions compared with the SRBFs using one-stage LHD methods. SRBF with S-SLE outperforms the other competitors in terms of convergence performance. Moreover, SRBF incorporated with S-SLE requires the fewest $Nfe$ for all the problems, which demonstrates that the S-SLE method

| Test function | $n$ | Theoretical global optimum | $f^*$ | $Nfe$ | $f^*$ | $Nfe$ | $f^*$ | $Nfe$ |
|---------------|-----|-----------------------------|-------|-------|-------|-------|-------|-------|
| SC            | 2   | −1.032                      | 1.030 | 65.5  | 0.918 | 91.0  | 1.032 | 32.3  |
| BR            | 2   | 0.398                       | 0.399 | 70.9  | 0.406 | 60.4  | 0.398 | 31.5  |
| PK            | 2   | −6.551                      | −6.225 | 54.5 | −4.540 | 51.1  | −6.551 | 32.1  |
| RS            | 2   | −2.000                      | −1.746 | 64.4 | −1.743 | 85.6  | −1.900 | 47.3  |
| GN            | 2   | 0.000                       | 1e−2  | 79.0  | 5.4e−1 | 100.6 | 1.7e−4 | 52.3  |
| HN            | 6   | −3.320                      | −3.107 | 89.3 | −2.934 | 188.2 | −3.321 | 69.4  |
| HD1           | 10  | 0.000                       | 0.447 | 275.8 | N/A   | N/A   | 0.064  | 322.2 |
| ICED          | 4   | 55.67                       | 51.470 | 187.8 | 51.646 | 220.3 | 52.132 | 129.9 |

Note: lhsdesign = MATLAB function; SLE = successive local enumeration; SC = six-hump camel-back function; BR = Branin function; PK = peaks function; RS = Rastrigin function; GN = Griewank function; HN = Hartman function; HD1 = SUR-T1-14; ICED = internal combustion engine design problem.
Table 3. Optimization results of sequential radial basis functions (SRBFs) using different sequential maximin Latin hypercube design (LHD) methods.

| Test function | n | Quasi-LHD | MIPT | MCSR | S-SLE |
|---------------|---|-----------|------|------|-------|
|               |   | Theoretical global optimum | $f^*$ | $N_{fe}$ | $f^*$ | $N_{fe}$ | $f^*$ | $N_{fe}$ | $f^*$ | $N_{fe}$ | $f^*$ | $N_{fe}$ |
| SC            | 2 | -1.032 | -1.032 | 38.0 | -1.032 | 36.0 | -1.032 | 35.3 | -1.032 | 32.3 |
| BR            | 2 | 0.398 | 0.398 | 36.1 | 0.398 | 33.5 | 0.398 | 32.0 | 0.398 | 31.5 |
| PK            | 2 | -6.551 | -6.551 | 34.4 | -6.551 | 33.4 | -6.551 | 35.7 | -6.551 | 32.1 |
| RS            | 2 | -2.000 | 1.897 | 49.8 | -1.854 | 49.9 | -1.859 | 53.4 | -1.900 | 47.3 |
| GN            | 2 | 0.000 | 5.0e-4 | 63.1 | 0.9e-4 | 57.3 | 1.5e-4 | 57.2 | 1.7e-4 | 52.3 |
| HN            | 6 | -3.320 | 3.314 | 81.7 | -3.320 | 79.4 | -3.321 | 71.7 | -3.321 | 69.4 |
| HD1           | 10 | 0.000 | 0.198 | 340.6 | 0.118 | 328.6 | 0.056 | 346.4 | 0.064 | 322.2 |
| ICED          | 5 | 55.67 | 52.178 | 157.6 | 51.618 | 144.7 | 50.968 | 145.0 | 52.132 | 129.9 |

Note: MIPT = mc-intersite-proj-th; MCSR = Monte Carlo method and space reduction strategy; S-SLE = sequential-successive local enumeration; SC = six-hump camel-back function; BR = Branin function; PK = peaks function; RS = Rastrigin function; GN = Griewank function; HN = Hartman function; HD1 = SUR-T1-14; ICED = internal combustion engine design problem.

has an advantage of improving optimization efficiency beyond the other three sequential LHD methods. The favourable space-filling and projective performance of S-SLE means that SRBF incorporated with S-SLE has the best convergence capability and highest efficiency. Box plots of $f^*$ and $N_{fe}$ for those testing problems obtained by SRBFs using four sequential LHD methods are shown in Figures B1 and B2 (Appendix A), respectively. As can be seen from Figures B1 and B2, the variations in $f^*$ and $N_{fe}$ obtained by SRBF using S-SLE were also smaller than those from SRBFs using the other sequential LHD methods. The smaller variations in SRBF with S-SLE are due to the unique deterministic sampling feature of the S-SLE method. Thus, the S-SLE method also improves the robustness of metamodel-based optimization. Hence, S-SLE has been demonstrated to show benefits by improving the global convergence capability, efficiency and robustness of sequential metamodel-based optimization algorithms, since more accurate metamodels can be constructed using S-SLE sequential samples with better space-filling and projective properties.

As pointed out by Jones et al. (1998), matrix ill-condition or even singularity may occur when newly added samples are too close to existing ones. That may result in poor accuracy of the updated metamodel and even optimization failure. Therefore, it is crucial for MBDO algorithms to prevent that situation. The proposed S-SLE has another merit in avoiding matrix ill-condition in metamodeling. To illustrate this merit, seven benchmark problems were solved by several SRBF variants using different LHD methods (including lhsdesign, SLE, TPLHD, Quasi-LHD, MIPT, MCSR and S-SLE) for sequential sampling. All of them were performed 100 times on each problem. Table 4 summarizes the occurrence numbers of matrix ill-condition for various LHD methods. Note that in this work, a matrix is regarded to be ill-conditioned if its reciprocal condition number is smaller than 1e-15. As can be seen from Table 4, the use of S-SLE and other sequential LHD methods significantly reduced the frequency of matrix ill-condition occurring in SRBF optimization compared with other one-stage OLHD methods. In fact, the S-SLE and MCSR methods again show slightly better performance than Quasi-LHD and MIPT in avoiding matrix ill-condition. Note that if the region of interest becomes too small, sequential maximin LHD methods are still likely to produce extremely crowded samples, which may finally cause matrix ill-condition. That is why a single occurrence of matrix ill-condition was found in SRBF optimization using S-SLE. Therefore, move-limit-based MBDO algorithms should also restrict the minimum size of the region of interest to avoid this numerical difficulty. Moreover, owing to worse space-filling performance, the Quasi-LHD yielded a larger frequency of matrix ill-condition than the other three sequential LHD methods.
Table 4. Summary of occurrence numbers of matrix ill-condition in sequential radial basis function (SRBF) optimization.

| Test function | lhsdesign | SLE | TPLHD | Quasi-LHD | MIPT | MCSR | S-SLE |
|---------------|-----------|-----|-------|-----------|------|------|-------|
| SC            | 4         | 15  | 12    | 0         | 0    | 0    | 0     |
| BR            | 5         | 12  | 12    | 1         | 0    | 0    | 0     |
| PK            | 4         | 25  | 20    | 1         | 0    | 0    | 0     |
| RS            | 8         | 15  | 15    | 2         | 1    | 0    | 0     |
| GN            | 4         | 32  | 14    | 2         | 2    | 0    | 0     |
| HN            | 2         | 2   | 2     | 1         | 0    | 0    | 0     |
| ICED          | 1         | 2   | 2     | 0         | 0    | 0    | 0     |

Note: lhsdesign = MATLAB function; SLE = successive local enumeration; TPLHD = translational propagation algorithm-based fast optimal Latin hypercube design; LHD = Latin hypercube design; MIPT = mc-intersite-proj-th; MCSR = Monte Carlo method and space reduction strategy; S-SLE = sequential-successive local enumeration; SC = six-hump camel-back function; BR = Branin function; PK = peaks function; RS = Rastrigin function; GN = Griewank function; HN = Hartman function; ICED = internal combustion engine design problem.

5. Conclusions

In this article, a novel deterministic sequential maximin LHD method (i.e. S-SLE) was proposed for metamodel-based optimization. The favourable projective property of the new samples was accomplished by using the proposed mesh-mapping algorithm. For pursuing better space-filling uniformity, S-SLE employed successive local enumerations to determine sequential samples by maximizing the minimal distance between new samples and existing points. Comparative studies with several existing LHD methods have been carried out to validate the performance of S-SLE. The comparison results demonstrated well the appealing merits of S-SLE. First, the sequential samples generated by S-SLE have better space-filling and projective properties, and the overall performance of S-SLE is also better than Quasi-LHD and MIPT in generating high-quality sequential samples, and shows comparable performance with MCSR. Secondly, as a deterministic sequential sampling method, for the given existing points, S-SLE always produces an identical sequential sample set without being affected by any stochastic factors. Besides, the flexible S-SLE is naturally applicable to multiple-stage sequential sampling applications. Finally, the proposed S-SLE can evidently improve the overall performance of sequential metamodel-based optimization, and reduce the frequency of numerical difficulty in metamodeling caused by overcrowded samples.

However, a major limitation of S-SLE has also been found in this work. As dimensionality grows, the local enumeration process becomes more and more computationally expensive, which significantly decreases the sampling efficiency of the S-SLE method for high-dimensional applications. In further work, more efficient integer programming algorithms will be tested to perform the local search to maximize the minimal distance between samples.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work is sponsored by the National Natural Science Foundation of China [grant nos 51105040 and 11372036], Aeronautical Science Foundation of China [grant no. 2011ZAZ72003], Excellent Young Scholars Research Fund of Beijing Institute of Technology [grant no. 2010Y0102] and Foundation Research Fund of Beijing Institute of Technology [grant no. 20130142008].
Supplemental material available online

Appendix A. Formulations of benchmark problems
Appendix B. Box plots of the optimization results for the benchmark problems
Supplemental material for this article can be accessed at http://dx.doi.org/10.1080/0305215X.2015.1081518 (also available at http://1drv.ms/1MBme4d).

References

Alexandrov, N. M., J. E. Dennis, R. M. Lewis, and V. Torczon. 1998. “A Trust-Region Framework for Managing the Use of Approximation Models in Optimization.” *Structural Optimization* 15 (1): 16–23. doi:10.1007/BF01197433

van Beers, W. C. M., and J. P. C. Kleijn. 2008. “Customized Sequential Designs for Random Simulation Experiments: Kriging Metamodeling and Bootstrapping.” *European Journal of Operational Research* 186 (3): 1099–1113. doi:10.1016/j.ejor.2007.02.035

Crombecq, K., E. Laermans, and T. Dhaene. 2011. “Efficient Space-Filling and Non-Collapsing Sequential Design Strategies for Simulation-Based Modeling.” *European Journal of Operational Research* 214 (3): 683–696. doi:10.1016/j.ejor.2011.05.032

Forrester, A. I. J., and A. J. Keane. 2009. “Recent Advances in Surrogate-Based Optimization.” *Progress in Aerospace Sciences* 45 (1–3): 50–79. doi:10.1016/j.paerosci.2008.11.001

van Dam, E. R., B. Husslage, and D. den Hertog. 2007. “Maximin Latin Hypercube Designs in Two Dimensions.” *Operations Research* 55 (1): 158–169. doi:10.1287/opre.1060.0317

Jin, R., W. Chen, and A. Sudjianto. 2002. “One Sequential Sampling for Global Metamodeling in Engineering Design.” ASME 2002 Design Engineering Technical Conferences, Montreal, Canada, DETC 2002/DAC-34092.

Jin, R., W. Chen, and A. Sudjianto. 2005. “An Efficient Algorithm for Constructing Optimal Design of Computer Experiments.” *Journal of Statistical Planning and Inference* 134 (1): 268–287.

Johnson, M. E., L. M. Moore, and D. Ylvisaker. 1990. “Minimax and Maximin Distance Designs.” *Journal of Statistical Planning and Inference* 26 (2): 131–148.

Jones, D. R., M. Schonlau, and W. J. Welch. 1998. “Efficient global optimization of expensive black-box functions.” *Journal of Global Optimization* 13 (4): 455–492.

Kim, B., Y. Lee, and D. H. Choi. 2009. “Construction of the Radial Basis Function Based on a Sequential Sampling Approach Using Cross-Validation.” *Journal of Mechanical Science and Technology* 23 (12): 3357–3365. doi:10.1007/s12206-009-1014-z

Li, Y., L. Liu, T. Long, and W. Dong. 2013. “Metamodel-Based Global Optimization Using Fuzzy Clustering for Design Space Reduction.” *Chinese Journal of Mechanical Engineering* 26 (5): 928–939. doi:10.3901/CJME.2013.05.928

Liu, H., S. Xu, and X. Wang. 2015. “Sequential Sampling Designs Based on Space Reduction.” *Engineering Optimization* 47 (7): 867–884.

Long, T., L. Liu, L. Peng, and Y. Li. 2012. “Aero-Structure Coupled Optimization of High Aspect Ratio Wing Using Enhanced Adaptive Response Surface Method.” 12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference, 17–19 Sep. 2012, Indianapolis, Indiana.

Morris, M. D., and T. J. Mitchell. 1990. “Exploratory Designs for Computer Experiments.” *Journal of Statistical Planning and Inference* 31 (3): 381–402. doi:10.1016/0378-3758(94)00035-T

Peng, L., L. Liu, T. Long, and W. Yang. 2014. “An Efficient Truss Structure Optimization Framework Based on CAD/CAE Integration and Sequential Radial Basis Function Metamodel.” *Structural and Multidisciplinary Optimization* 50 (2): 329–346. doi:10.1007/s00158-014-1050-x

Sacks, J., W. J. Welch, T. J. Mitchell, and H. P. Wynn. 1989. “Design and analysis of computer experiments.” *Statistical Science* 4 (4): 409–423.

Simpson, T. W., A. J. Booker, D. Ghosh, A. A. Giunta, P. N. Koch, and R. J. Yang. 2004. “Approximation Methods in Multidisciplinary Analysis and Optimization: A Panel Discussion.” *Structural and Multidisciplinary Optimization* 27 (5): 302–313. doi:10.1007/s00158-004-0389-9

Viana, F. A. C. 2011. “SURROGATES Toolbox User’s Guide.” Version 3.0, available at http://sites.google.com/site/felipeacviana/surrogatestoolbox

Viana, F. A. C., T. W. Simpson, V. Balabanov, and V. Toropov. 2014. “Metamodeling in Multidisciplinary Design Optimization: How Far Have We Really Come?” *AIAA Journal* 52 (4): 670–690. doi:10.2514/1.J052375.

Viana, F. A. C., G. Venter, and V. Balabanov. 2010. “An Algorithm for Fast Optimal Latin Hypercube Design of Experiments.” *International Journal for Numerical Methods in Engineering* 82 (2): 135–156.

Wagner, T. C., and P. Y. Papalambros. 1991. “Optimal Engine Design Using Nonlinear Programming and the Engine System Assessment Model.” Ford Motor Company, Technical Report No. SR-91–154.

Wang, G. G. 2003. “Adaptive Response Surface Method—A Global Optimization Scheme for Approximation-Based Design Problems.” *Engineering Optimization* 35 (6): 707–733. doi:10.1080/0305215031040940940
Wang, G. G., and S. Shan. 2007. “Review of Metamodeling Techniques in Support of Engineering Design Optimization.” *Journal of Mechanical Design* 129 (4): 370–380. doi:10.1115/1.2429697

Wang, G. G., and T. W. Simpson. 2004. “Fuzzy Clustering Based Hierarchical Metamodeling for Space Reduction and Design Optimization.” *Engineering Optimization* 36 (3): 313–335. doi:10.1080/03052150310001639911

Wang, L. Q., S. Q. Shan, and G. G. Wang. 2004. “Mode-Pursuing Sampling Method for Global Optimization on Expensive Black-Box Functions.” *Engineering Optimization* 36 (4): 419–438. doi:10.1080/03052150410001686486.

doi:10.1080/03052150410001686486

Xiong, F., Y. Xiong, W. Chen, and S. Yang. 2009. “Optimizing Latin Hypercube Design for Sequential Sampling of Computer Experiments.” *Engineering Optimization* 41 (8): 793–810. doi:10.1080/03052150902852999

Ye, K. Q., W. Li, and A. Sudjianto. 2000. “Algorithmic Construction of Optimal Symmetric Latin Hypercube Designs.” *Journal of Statistical Planning and Inference* 90 (1): 145–159. doi:10.1016/s0378-3758(00)00105-1.

doi:10.1016/S0378-3758(00)00105-1

Zhu, H. G., L. Liu, L. Long, and L. Peng. 2012. “A Novel Algorithm of Maximin Latin Hypercube Design Using Successive Local Enumeration.” *Engineering Optimization* 44 (5): 551–564.