Heuristic Search Algorithms for Constructing Optimal Latin Hypercube Designs

Anamai Na-udom and Jaratsri Rungrattanaubol

Abstract Computer simulated experiments (CSEs) have been extensively used to explore the relationship between input variables and output response in science and engineering applications. Normally, CSE process is time consuming and computationally expensive to run and the output response from computer simulated experiments is deterministic. Consequently the space filling designs, which focus on spreading design points over a design space, are necessary. Latin hypercube designs (LHD) are widely used in the context of CSE. The optimal LHD for a given dimension of problem is constructed by using search algorithms under pre-specified optimality criteria. This paper proposes the methods to enhance the performance of search algorithms which have been widely used in the context of CSE. The results indicate that the proposed enhancement method can improve the performance of the search algorithms for constructing the optimal LHD.

Keywords Optimal designs · Search algorithm · Computer simulated experiments · Latin hypercube design

1 Introduction

In the past three decades, computer simulated experiments (CSE) have replaced classical experiments to investigate physical complex phenomena, especially when classical (physical) experiments are not feasible. For example, the use of reservoir simulator to predict ultimate recovery of oil, the use of finite element codes to predict

A. Na-udom
Faculty of Science, Department of Mathematics, Naresuan University, Phitsanulok, Thailand
e-mail: anamain@nu.ac.th

J. Rungrattanaubol (✉)
Faculty of Science, Department of Computer Science and Information Technology, Naresuan University, Phitsanulok, Thailand
e-mail: jaratsrir@nu.ac.th

P. Meesad et al. (eds.), *Recent Advances in Information and Communication Technology 2016*, Advances in Intelligent Systems and Computing 463,
DOI 10.1007/978-3-319-40415-8_18

behavior of metal structure under stress, and so on [1]. The nature of computer simulated experiments is deterministic [2]; hence identical settings of input variables always produce an identical set of output response. Consequently, space filling designs that aim to spread the design points over a region of interest are required. The most popular class of space filling design in the context of computer simulated experiments is latin hypercube design (LHD). LHD design was originally proposed by Mckay and co-workers [3] in 1979. The optimal LHD can be constructed through combinatorial methods (non-search algorithm) [4] or through search algorithms [5, 6]. The former method generates design with good design properties but it is restricted in terms of a design size. For example methods proposed by Butler [4] are limited to a design size of a prime number. The latter method is based largely on improving design by exchanging between the pairs of design points. Exchange algorithms can be time consuming to implement, however, the generated design are flexible and straightforward. The CSEs are usually complex and consist of many input variables to investigate [7] and hence a large number of design runs are required to estimate the parameters corresponding to the factors of interest in the model. For example, if the problem of interest consists of d input variables and n number of runs, the total number of all possible LHD is $(n!)^d$. Obviously this number explodes exponentially as the values of n and d increase; hence the full space of LHD cannot be explored. In this case we need search algorithms to lead us to a good design with respect to an optimality criterion. The key idea of all existing search algorithms is to use some kinds of exchange procedures to move towards the better designs.

The search based approach for selecting a design is implemented by combining search algorithms and the optimality criterion [8]. For example, Morris and Mitchell [5] adopted a version of Simulated Annealing algorithms (SA) to search for optimal LHDs with respect to ϕ_p criterion. Li and Wu [8] proposed a columnwise-pairwise algorithm (CP) with respect to the D efficiency criterion. It was reported that CP is very simple and easy to implement. Ye and his co-workers [6] adapted CP algorithm to search for symmetric LHD under various optimality criteria such as entropy and ϕ_p criteria. Park [9] proposed a row-wise element exchange algorithm along with IMSE and entropy criteria. Leary et al. [10] adapted CP and SA algorithms to construct the optimal designs within the orthogonal-array based Latin hypercube class by using the ϕ_p criteria. Jin et al. [11] developed an enhanced stochastic evolutionary algorithm (ESE) to search for the best design considering various optimality criteria such as a maximin distance criterion, ϕ_p criterion and entropy criterion. ESE has received wide attention from researchers due to its performance in constructing the optimal LHD. Liefvendahl and Stocki [12] applied a version of Genetic algorithm (GA) to search for the optimal LHD considering ϕ_p and a maximin distance criterion. A similar work can be found in [13] as the authors applied GA for constructing maximin designs. Grosso et al. [14] used the iterated local search algorithm and SA in constructing the optimal LHD under maximin distance and ϕ_p criterion. Vianna et al. [15] proposed the algorithm for fast optimal LHD by using the idea of seed design under maximin distance and ϕ_p criterion. Husslage and Rennen [16] proposed the method to construct the optimal LHD using different starting points. Due to the popularity of SA and ESE

along with ϕ_p criteria, this paper presents the enhancement method to improve the capability of SA and ESE under ϕ_p criterion. In the following sections, we describe designs and optimality criteria, followed by search algorithms and its modification, results and conclusion, respectively.

2 Latin Hypercube Design and Optimality Criteria

2.1 Latin Hypercube Design

LHD is a matrix ($\boldsymbol{X}$), which obtains n rows and d columns where n is the number of runs and d is the number of input variables. LHD can be constructed based on the idea of stratified sampling, which can ensure all subregions in the space are sampled with equally probability. A Latin hypercube sampling has

$$X_{ij} = \frac{\pi_{ij} - U_{ij}}{n}, \tag{1}$$

where $\pi_{i1}, \pi_{i2}, \ldots, \pi_{id}$ are independent random permutation of (1, 2, …, n) and U_{ij} are n x d values of *i.i.d.* uniform $U[0,1]$ random variables independent of the π_{ij}. In practice, LHD can be easily generated by a random permutation of each column which contains the levels (1, 2 ,…, n). Then the d columns are combined together to form the design matrix $\boldsymbol{X}$. LHD can ensure uniform coverage of each input variable from a different single dimension. This shows a benefit of LHD on deterministic computer experiments.

2.2 The ϕ_P Optimality Criterion

Morris and Mitchell [5] proposed a modification of maximin distance criterion to search for the optimal design. For a given design X, the Euclidean intersite distance between any two design points can be calculated from

$$d(x_i, x_j) = \left[\sum_{k=1}^{d} (x_{ik} - x_{jk})^2 \right]^{1/t} \tag{2}$$

By using (2), all intersite distances for every pairs of design points are calculated and can be expressed in a symmetric matrix. Let Euclidean distance list $d_1, d_2, \ldots, d_m$ be the distinct elements list from the smallest to largest, and also define index list $(J_1, J_2, \ldots, J_m)$ which J_j is the number of pairs of sites in the design separated by distance d_j. Thus X is a maximin design if among available designs, it maximizes d_j while J_j is minimized. The scalar criterion can be expressed as

$$\phi_p = [\sum_{j=1}^{m} J_j d_j^{-p}], \tag{3}$$

where p is a positive integer, J_j and d_j specified from X. In this study, the adaptive form of ϕ_p [11] which is simpler than (3) to implement is considered

$$\phi_p = \left[\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{d_{ij}^p}\right]^{\frac{1}{p}} \tag{4}$$

After ϕ_p value has been calculated, a design that minimizes ϕ_p is considered as an optimal design.

3 Search Algorithms

There are many search algorithms for constructing optimal LHD. This section will firstly discuss two search algorithms Stimulated Annealing (SA) and Enhanced Stochastic Evolutionary (ESE), and then explain how to modify and improve each algorithm for better efficiency.

3.1 Simulated Annealing (SA) Algorithm

SA is based on the analogy between the simulation of annealing solids and the problem solving of large combinatorial optimization problems [9]. Morris and Mitchell [5] adapted SA to construct optimal LHD using ϕ_p optimality criterion. LHD minimizing ϕ_p value is reserved as the best optimal LHD in the search space. The entire process can be described in Algorithm 1.

Algorithm 1: Simulated Annealing (SA)

1. Initialize I_{max}, t_0 and C_t
2. Set X to a random LHD, $X_{best} = X$, $t = t_0$
3. Set $I = 1$, $Label = 0$
4. Construct X_{try} by randomly element exchange in X
5. If $\phi_p(X) - \phi_p(X_{try}) \geq tl$ or accept with probability $e^{-\left[\phi_p(X_{try})-\phi_p(X)\right]/t}$ then $X = X_{try}$, $Label = 1$
6. If $\phi_p(X_{try}) < \phi_p(X_{best})$ then $I = 1$, $X_{best} = X_{try}$
   Else $I = I + 1$
7. If $I < I_{max}$ then goto 4
   Else $t = t * C_t$
8. If $Label = 1$ then goto 3
   Else report X_{best}

SA performs searching by obtaining a new LHD (X_{try}) via randomly element-exchange and update X with a better LHD or a worse LHD with satisfying probability, then update X_{best} with a better LHD. The probability of accepting a worse design is controlled by a value of temperature t, a chance of worse LHD acceptance decreases by a cooling system ($t = t * C_t$). SA has several parameters such as t_0, I_{max} and C_t. The discussion of setting SA parameters for LHD construction and how well SA can perform in terms of moving away from a local optimum value of ϕ_p can be seen in [5]. In this study, we use a heuristic method to find the best set of parameters to use in SA as presented in [17].

This paper focuses on improving the efficiency of SA to construct optimal LHD. From Eq. (4), it can be obviously seen that time for ϕ_p calculation mainly depends on the number of run (n), hence it is approximate to the Euclidean distance matrix calculation (n^2). The search process takes a long time mainly because of ϕ_p calculation. Jin et al. [11] has proposed an optimized way to calculate ϕ_p when element-exchange is assigned to LHD. The element-exchange is a swap operation between two selected points. Hence, when exchanging points between rows i_1 and i_2 within column k ($x_{i_1k} \leftrightarrow x_{i_2k}$), only elements in rows i_1 and i_2, and columns i_1 and i_2 in the distance matrix are changed. The ϕ_p of LHD after element-exchange can be calculated with a linear time as follows.

For any $1 \le j \le n$ and $j \ne i_1, i_2$ let

$$s(i_1, i_2, k, j) = \left|x_{i_2k} - x_{jk}\right|^t - \left|x_{i_1k} - x_{jk}\right|^t \tag{5}$$

then

$$d'_{i_1j} = d'_{ji_1} = \left[d^t_{i_1j} + s(i_1, i_2, k, j)\right]^{1/t} \tag{6}$$

and

$$d'_{i_2j} = d'_{ji_2} = \left[d^t_{i_2j} + s(i_1, i_2, k, j)\right]^{1/t} \tag{7}$$

Thus new ϕ_p is computed by

$$\phi'_p = \left[\phi_p^p + \sum_{1 \le j \le n, j \ne i_1, i_2} \left[(d'_{i1j})^{-p} - (d_{i1j})^{-p}\right] + \sum_{1 \le j \le n, j \ne i_1, i_2} \left[(d'_{i2j})^{-p} - (d_{i2j})^{-p}\right]\right]^{1/p} \tag{8}$$

As shown in (5)–(8), when performing element exchange, only some rows and columns will be updated, hence there is no need to reconstruct the entire distance matrix to calculate ϕ_p, hence the complexity of ϕ_p calculation reduces to a linear time. Therefore, after this modification, the modified SA (MSA) performs effectively by reducing time complexity especially in ϕ_p calculation.

3.2 Enhanced Stochastic Evolutionary (ESE) Algorithm

Jin et al. [11] proposed an algorithm called enhanced stochastic evolutionary (ESE) to construct an optimal design for CSE. The algorithm performs searching in 2 steps, a local search called inner loop and updating a global best and fine tuning probability of accepting a worse design called outer loop. The inner loop performs a local search by constructing a set of LHD and selecting the optimal LHD from the set. The set contains J LHDs formed by element-exchanging at column i mod d of X. If the selected design is better or not better but has good enough probability then the local optimal design (X) will be updated.

Algorithm 2: Enhancement of Stochastic Evolutionary (ESE)

1. Initialize Th, J, M
2. Set X to a random LHD, $X_{best} = X$
3. Set $X_{old\_best} = X_{best}$, $i = 0$, $n_{acpt} = 0$ and $n_{imp} = 0$
4. Randomly create J distinct LHDs by element exchanging X at column(i mod d)
5. Select the best LHDs from J distinct LHDs and assign it to X_{try}
6. If $\phi_p(X) - \phi_p(X_{try}) \geq tl$ or $\phi_p(X_{try}) - \phi_p(X) \leq Th * random(0,1)$ then
   $X = X_{try}$, $n_{acpt} = n_{acpt} + 1$
   If $\phi_p(X_{try}) < \phi_p(X_{best})$ then $X_{best} = X_{try}$, $n_{imp} = n_{imp} + 1$
7. If $i < M$ then $i = i + 1$, goto 4
   Else goto 8
8. If $\phi_p(X_{old\_best}) - \phi_p(X_{best}) \geq tl$ then $flag_{imp} = 1$
   Else $flag_{imp} = 0$
9. If $flag_{imp} = 1$ then
   If $n_{acpt}/M > \beta_1$ and $n_{imp}/M < n_{acpt}/M$ then
   $Th = Th * \alpha_1$
   Else if $n_{acpt}/M > \beta_1$ and $n_{imp}/M = n_{acpt}/M$ then
   $Th = Th$
   Else
   $Th = Th / \alpha_1$
   Else
   If $n_{acpt}/M \geq \beta_1$ and $n_{acpt}/M \leq \beta_2$ and step=0 then
   If step=0 or step=1 then $Th = Th / \alpha_3$
   Else If step=2 then $Th = Th * \alpha_2$
   Else If $n_{acpt}/M \leq \beta_1$ then
   $Th = Th / \alpha_3$
   step =1
   Else If $n_{acpt}/M \geq \beta_2$ then
   $Th = Th * \alpha_2$
   Step =2
10. If $\phi_p(X_{old\_best}) - \phi_p(X_{best}) \geq tl$ then $X_{best} = X_{old\_best}$, $i_{out} = 1$
    Else $i_{out} = i_{out} + 1$
11. If $i_{out} < Max$ then Goto 3
    Else Report X_{best}

From Algorithm 2, the inner loop performs in 4–7, and repeats with the maximum loop (M). The X_{best} and X are updated with the acceptance criteria. The outer loop controls the process by updating the value of temperature Th. Unlike SA, the process of updating the temperature (Th) is not fixed, but is controlled by the performance of searching in terms of the inner loop improvement by the number of improvement (n_{imp}) and number of acceptance (n_{acpt}). There are two processes of updating Th called improving process and exploration process. The process is described in 9, when X_{best} get improved in the inner loop and better than the previous best design (X_{old_best}), the improving process is active otherwise exploration process is active.

In improving process ($flag_{imp} = 1$), Th is adjusted based on the performance in a local best LHD search by considering an acceptance ratio (n_{acpt}/M) and the improvement ratio in (n_{imp}/M), β_1 and β_2 are cutting point values for updating Th, where $0 < \beta_1 < 1$. Jin et al. [4] recommended β_1 to be 0.1. If the improvement ratio is greater than β_1, Th is increased by α_1 ($Th = Th/\alpha_1$), else if the improvement ratio is lower than the acceptance ratio Th is decreased by α_1 ($Th = Th * \alpha_1$), otherwise unchanged, where $0 < \alpha_1 < 1$ and $\alpha_1 = 0.8$ as suggested by Jin et al. [4].

In exploration process ($flag_{imp} = 0$), when a local best LHD performs worse, Th is adjusted to help a search process moving away from a local optimal design by considering only the accept ratio. If the acceptance ratio is between β_1 and β_2 where $0 < \beta_1 < \beta_2 < 1$, Th is increased by α_2. If the acceptance ratio is greater than β_2, Th is decreased by α_3, where $0 < \alpha_2 < \alpha_3 < 1$, $\alpha_2 = 0.9$ and $\alpha_3 = 0.7$, Th is rapidly increased, this means more worse designs could be accepted, then Th is decreased slowly for searching better design after moving away from a local optimal design. Jin et al. [11] recommended 0.1 for β_1 and 0.8 for β_2.

3.3 *Modified Enhancement of Stochastic Evolutionary (MESE)*

This section we present the enhancement method on ESE. The modified version is called MESE. We combine the advantage of SA (i.e. local search process) and the advantage of ESE (i.e. global search process) to improve the search process. MESE still contains 2 nested loops. The outer loop is similar to ESE except a stopping rule and a new variable X_{gbest} to record the best so far LHD, while the inner loop is modified as describe in Algorithm 3. The maximum number of cycles used is replaced by the following condition. If X_{best} is not improved from the global best design (X_{gbest}) for δ consecutive times, the search process is terminated. In this study, we set $\delta = 40$ [6].

We have modified ESE by combining a part of SA into the modified ESE (MESE). Algorithm 3 explains this modification.

Algorithm 3: Modified Enhancement of Stochastic Evolutionary (MESE)

1. Initialize Th, J, M, Max, C_{max}
2. Set X to a random LHD, $X_{best} = X$, X_{gbest}, $i_{out} = 0$
3. Set $X_{old\_best}=X_{best}$, $i=0$, $n_{acpt}=0$, $n_{imp}=0$, $j=0$ and $X_{try}=X$
4. $X_{tmp}=X_{try}$, $c=0$
5. X_{tmp}=Random element exchanging X at column(i mod d)
6. If $\phi_p(X_{tmp}) - \phi_p(X_{try}) \geq tl$ then $X_{try} = X_{tmp}$
   If $c<J$ then $c=c+1$, goto 5
   Else goto 7
   Else goto 5
7. If $\phi_p(X_{try}) - \phi_p(X) \geq tl$ or $\phi_p(X_{try}) - \phi_p(X) \leq Th*random(0,1)$ then
   $X = X_{try}$, $n_{acpt} = n_{acpt}+1$
   If $\phi_p(X_{try}) < \phi_p(X_{best})$ then $X_{best}=X_{try}$, $n_{imp}=n_{imp}+1$, $j=1$
   Else $j = j + 1$
   Else goto 8
8. If $i<M$ OR $j<C_{max}$ then $i = i+1$, goto 4
   Else goto 9
9. Perform step 8 and step 9 in Algorithm 2
10. If $\phi_p(X_{best}) - \phi_p(X_{gbest}) \geq tl$ then $\phi_p(X_{gbest}) = \phi_p(X_{best})$, $i_{out}=1$
   Else $i_{out} = i_{out}+1$
11. If $i_{out} < Max$ then goto 3
   Else report X_{gbest}

In Algorithm 3, the major enhancement was made in the inner loop; the modification is on 4–6 and 7. Instead of generating J distinct LHDs at one time and select the best one, we modify it by constructing one LHD (X_{tmp}) at a time and keep the best one (X_{try}) for J iterations. This change can decrease the computational time complexity since we keep the optimal LHD while generating it, instead of obtaining the entire distinct J LHDs and deciding the most optimal one. In this study the parameters J is set to be ${}^{n}C_2/5$ but not larger than 50, and the parameter M is in a range of $2 * {}^{n}C_2 * d/J \leq M \leq 100$. The tolerance level ($tl$) is set to 0.0001, from the empirical study, the smaller value does not improve the search process. $C_{max} = d * 10 + 10$ and $Max = 40$. All simulation studies presented in this paper were performed using R program.

4 Result and Discussion

The values of ϕ_p at the termination step of MSA, ESE and MESE for each dimension of problems are presented in Table 1. Each case was repeated for 10 times to consider the effect of different starting points. The descriptive statistics (max, min, mean and sd.) on the ϕ_p values obtained from each search technique are

Table 1 Performance of MSA, ESE and MESE

		ϕ_p				Time (s.)	No. exchange
n × d	method	max	min	mean	sd.	average	average
9 × 2	MSA	4.273538	4.273538	**4.273538**	1.48E-12	16.140	47044.7
	ESE	4.344617	4.273538	4.301970	0.034821	2.854	5760.0
	MESE	4.344617	4.273538	4.294007	0.031286	**1.972**	**5644.8**
19 × 3	MSA	4.936365	4.898022	**4.916390**	0.013472	103.194	140022.5
	ESE	4.935001	4.901203	4.922769	0.010908	53.183	**41040.0**
	MESE	4.958163	4.914072	4.926576	0.011805	**30.942**	42160.0
99 × 7	MSA	5.756299	5.750880	5.753385	0.001473	769.902	193111.8
	ESE	5.745721	5.740989	5.742462	0.001387	954.711	200000.0
	MESE	5.745267	5.740459	**5.742109**	0.001577	**800.993**	**199865.0**
129 × 8	MSA	5.904712	5.901620	5.902779	0.000759	1070.692	201891.2
	ESE	5.891672	5.887732	**5.889600**	0.001383	1217.445	200000.0
	MESE	5.891365	5.888284	5.890195	0.001029	**1063.358**	200000.0
201 × 10	MSA	6.182737	6.179070	6.181468	0.001015	1905.174	218859.0
	ESE	6.164494	6.162494	6.163368	0.000648	1885.965	200000.0
	MESE	6.164341	6.161576	**6.163364**	0.000721	**1717.370**	200000.0
451 × 15	MSA	6.777918	6.775912	6.777281	0.000589	5338.902	231194.8
	ESE	6.754473	6.753710	6.754034	0.000220	4646.026	200000.0
	MESE	6.754603	6.753482	**6.753898**	0.000321	**4404.563**	200000.0
801 × 20	MSA	7.274153	7.272498	7.273293	0.000546	11892.890	251293.7
	ESE	7.248591	7.247829	7.248093	0.000262	9035.538	200000.0
	MESE	7.248234	7.247825	**7.248006**	0.000136	**8880.095**	200000.0

presented. The results indicate that MSA, ESE and MESE perform similarly for small dimension of problem (i.e. d = 2 and 3) in terms of minimization of ϕ_p. Further, the standard deviation values displayed a slightly larger amount of variation over 10 replications in ESE and MESE than that of MSA. This indicates the consistency in the search process for MSA when different starting points are considered. When considering medium dimension of problems (i.e. d = 7, 8 and 10), ϕ_p values from ESE and MESE are slightly lower than MSA. In addition, the standard deviation values obtained from ESE and MESE are smaller than MSA. This indicates that the search process of ESE and MESE is more consistent when the search space is larger. For large dimensions of problem (i.e. d = 10 and 15), both of ESE and MESE perform better than MSA while MESE is slightly better than ESE in terms of minimization of ϕ_p values. Hence if the goal is concerned with a good space filling design property, either ESE or MESE can be used for constructing the optimal LHD.

The results of the performance (efficiency) for MSA, ESE and MESE algorithms are presented in terms of time and number of exchanges for each algorithm to reach the optimal ϕ_p values, as shown in Table 1. As mentioned before, for each dimension of problems the search algorithms are repeated for 10 times, hence all

values are presented as the average values. It can be clearly seen that MESE converges much faster than ESE and MSA as the time elapsed is less than that of ESE and MSA. When considering the number of exchange required in the search process, it is observed that number of exchange in MESE is less than the other two algorithms. The similar results are also observed in the case of medium and large dimension of problems as MESE converges much faster than MSA while it performs slightly better than ESE. This indicates that if time constraint is taken into account, MESE could be the better choice for constructing the optimal LHD designs.

5 Conclusion

This paper presents the method to enhance the SA and ESE algorithms in the construction of the optimal LHD. The major enhancement method appears in the calculation of ϕ_p criterion and the tolerance level setting in SA. For MESE, the enhancement is applied by using the combination of SA and ESE especially in the inner loop as shown in Algorithm 3. As presented in the result section, MESE performs better than ESE and MSA in terms of the design property achievement and the efficiency. Hence MESE would be recommended for the construction of optimal LHD for CSE. In order to extend the conclusion, other classes of design can be developed and collaborated with MESE to search for the best design in the class. Further, other types of search algorithm like Particle swarm optimization (PSO) or any type of clever algorithms can be further developed in constructing an optimal LHD. The validation of the approximation model accuracy developed from the obtained optimal LHD could also be further investigated in order to explore the relation of space filling property and prediction accuracy of the model.

References

1. Koehler, J., Owen, A.B.: Computer experiments. Handbook of Statistics, vol. 13, pp. 261–308. Elsevier Science, New York (1996)
2. Sacks, J., Welch, W.J., Mitchell, T.J., Wynn, H.P.: Design and analysis of computer experiments. Stat. Sci. **4**(4), 409–435 (1989)
3. Mackay, M.D., Beckman, R.J., Conover, W.J.: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics **21**, 239–246 (1979)
4. Butler, N.A.: Optimal and orthogonal latin hypercube designs for computer experiments. Biometrika **88**(3), 847–857 (2001)
5. Morris, M.D., Mitchell, T.J.: Exploratory design for computational experiments. J. Stat. Plann. Infer. **43**, 381–402 (1995)
6. Ye, K.Q., Li, W., Sudjianto, A.: Algorithmic construction of optimal symmetric latin hypercube designs. J. Stat. Plann. Infer. **90**, 145–159 (2000)
7. Bates, R.A., Buck, R.J., Riccomagno, E., Wynn, H.P.: Experimental design and observation for large systems. J. Roy. Stat. Soc. B **58**, 77–94 (1996)

8. Li, W., Wu, C.F.J.: Columnwise-pairwise algorithms with applications to the construction of supersaturated designs. Technometrics **39**, 171–179 (1997)
9. Park, J.S.: Optimal latin hypercube designs for computer experiments. J. Stat. Plann. Infer. **39**, 95–111 (1994)
10. Leary, S., Bhaskar, A., Keane, A.: Optimal orthogonal-array-based latin hypercubes. J. Appl. Stat. **30**(5), 585–598 (2003)
11. Jin, R., Chen, W., Sudjianto, A.: An efficient algorithm for constructing optimal design of computer experiments. J. Stat. Plann. Infer. **134**, 268–287 (2005)
12. Liefvandahl, M., Stocki, R.: Study on algorithms for optimization of latin hypercubes. J. Stat. Plann. Infer. **136**, 3231–3247 (2006)
13. Li, Z., Shigeru, N.: Maximin distance-lattice hypercube design for computer experiment based on genetic algorithm. IEEE Explore **2**, 814–819 (2001)
14. Grosso, A., Jamali, A., Locatelli, M.: Finding maximin latin hypercube designs by iterated local search heuristics. Eur. J. Oper. Res. **197**(2), 541–547 (2009)
15. Viana, F.A.C., Venter, G., Balanov, V.: An algorithm for fast optimal latin hypercube design of experiments. Int. J. Numer. Meth. Eng. **82**(2), 135–156 (2010)
16. Husslage, B.G.M., Rennen, G., van Dam, E.R., Hertog, D.D.: Space-filling Latin hypercube designs for computer experiments. Optim. Eng. **12**, 611–630 (2011)
17. Na-udom, A.: Experimental design methodology for modeling response from computer simulated experiments. Ph.D. thesis, Curtin University of Technology (2007)