Miscellanea

Construction of orthogonal Latin hypercube designs

BY FASHENG SUN, MIN-QIAN LIU

The Key Laboratory of Pure Mathematics and Combinatorics, School of Mathematical
Sciences, Nankai University, Tianjin 300071, China
sfxsfx2001@mail.nankai.edu.cn mqliu@nankai.edu.cn

AND DENNIS K. J. LIN

Department of Statistics, The Pennsylvania State University, University Park, Pennsylvania
16802, U.S.A.
dk15@psu.edu

SUMMARY

Latin hypercube designs have found wide application. Such designs guarantee uniform samples for
the marginal distribution of each input variable. We propose a method for constructing orthogonal Latin
hypercube designs in which all the linear terms are orthogonal not only to each other, but also to the quadratic
terms. This construction method is convenient and flexible, and the resulting designs can accommodate
many more factors than can existing ones.

Some key words: Computer experiment; Factorial design; Orthogonality; Second-order model.

1. INTRODUCTION

Many physical phenomena encountered in science and engineering are governed by a set of equations
that can only be solved by a computer. Because such models are mostly deterministic, computer
experiments require special designs, such as Latin hypercube designs, which possess equally spaced lev-
els. Orthogonality is an important criterion in choosing Latin hypercube designs. Ye (1998) presented
a method for constructing orthogonal Latin hypercube designs in which all the input factors have zero
correlation. On the basis of Ye’s procedure, Cioppa & Lucas (2007) augmented the number of factors
for some cases. Beattie & Lin (2004) presented a class of orthogonal Latin hypercube designs developed
from the rotation of $q$-level full factorial designs. Recently, Steinberg & Lin (2006) and Pang et al. (2009)
proposed methods to construct orthogonal Latin hypercube designs by means of rotating factorial designs,
while Bingham et al. (2009) and Lin et al. (2009) constructed many orthogonal or nearly orthogonal Latin
hypercube designs.

Orthogonal Latin hypercube designs ensure independence of estimates of linear effects when a first-
order model is fitted. However, a second-order model is needed when second-order effects are present. For
such cases, a Latin hypercube design must satisfy the following properties: (a) each column is orthogonal
to the others in the design; (b) the elementwise square of each column and the elementwise product of
every two columns are orthogonal to all columns in the design. The Latin hypercube designs constructed
by Ye (1998) possess these properties, but can accommodate only a few factors. In this paper, we propose a
method for constructing Latin hypercube designs with properties (a) and (b) that can accommodate many
more factors than existing ones.
2. The construction method

2.1. Orthogonal Latin hypercube designs with $2^{c+1} + 1$ runs and $2^c$ factors

We denote a Latin hypercube design with $n$ runs, rows, and $k$ factors, columns, by $L(n, k)$. For any integer $c \geq 1$, the construction algorithm for an orthogonal $L(2^{c+1} + 1, 2^c)$ can be illustrated as follows.

Step 1. For $c = 1$, let

$$ S_1 = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}, \quad T_1 = \begin{pmatrix} 1 & 2 \\ 2 & -1 \end{pmatrix}. $$

(1)

Step 2. For $c > 1$, define $S_c$ and $T_c$ as

$$ S_c = \begin{pmatrix} S_{c-1} & -S^*_{c-1} \\ S_{c-1} & S^*_{c-1} \end{pmatrix}, \quad T_c = \begin{pmatrix} T_{c-1} & -(T^*_{c-1} + 2^{c-1}S^*_{c-1}) \\ T_{c-1} + 2^{c-1}S_{c-1} & T^*_{c-1} \end{pmatrix}, $$

(2)

where the * operator works on any matrix with an even number of rows by multiplying the entries in the top half of the matrix by $-1$ and leaving those in the bottom half unchanged.

Step 3. An $L(2^{c+1} + 1, 2^c)$ can be obtained as

$$ L_c = \begin{pmatrix} T_c^T & 0_{2^c} & -T_c^T \\ \end{pmatrix}^T. $$

(3)

where $A^T$ denotes the transpose of $A$ and $0_{2^c}$ denotes the $2^c \times 1$ column vector with all elements zero.

Some theoretical properties of the proposed design can be stated as the following theorem. The proof is given in the Appendix.

**Theorem 1.** (i) The $T_c$ in (2) consists of rows and columns of permutations of the $2^c$ elements $1, \ldots, 2^c$, up to sign changes.

(ii) The $L_c$ in (3) is a Latin hypercube design $L(2^{c+1} + 1, 2^c)$ with properties (a) and (b).

2.2. Orthogonal $L(n, k)$s with $n = 2^{c+1}$ runs and $k = 2^c$ factors

The construction method described in the previous subsection can be modified to produce $L(2^{c+1}, 2^c)$s with properties (a) and (b). The construction algorithm is as follows.

Step 1'. As equation (1) for construction of $S_1$ and $T_1$.

Step 2'. As equation (2) for construction of $S_c$ and $T_c$.

Step 3'. Let $H_c = T_c - S_c/2$, $L_c = (H^T_c, -H^T_c)^T$.

Then from a similar argument given in the proof of Theorem 1 in the Appendix, it can easily be shown that $H^T_c H_c = h_c I_{2^c}$, where $h_c = 2^c(2^{2c+2} - 1)/12$. This leads to Theorem 2.

**Theorem 2.** The $L_c$ obtained in Step 3' is a Latin hypercube design $L(2^{c+1}, 2^c)$ with properties (a) and (b).

3. Comparisons with existing methods

Among the existing constructions for orthogonal Latin hypercube designs, only those proposed by Ye (1998) and Cioppa & Lucas (2007) can produce designs possessing both properties (a) and (b). For $n = 2^{c+1}$ or $2^{c+1} + 1$ runs, Ye's (1998) method can construct a design with at most $k = 2^c$ factors. The method of Cioppa & Lucas (2007) can generate a design with at most $k = \binom{c+1}{2} + 1$ factors, while the number of factors in the design constructed by our method is $k = 2^c$. Thus, our method allows a substantially larger number of factors than those of Ye (1998) and Cioppa & Lucas (2007), especially for large $c$. In fact, the number of factors $k$ in the design constructed by our method attains its maximum.
value among all the corresponding Latin hypercube designs satisfying both properties (a) and (b). This conclusion is based on the following theorem, which can be straightforwardly obtained.

**Theorem 3.** If \( L(n, k) = (l_{ij}) \) is a centered Latin hypercube design with properties (a) and (b), then \( k \leq \lfloor n/2 \rfloor \), where \( \lfloor x \rfloor \) is the integer part of \( x \).

**Acknowledgement**

This work was supported by the Program for New Century Excellent Talents in University and the National Natural Science Foundation of China. The authors thank the editor, the referees, and Dr. Nam-Ky Nguyen for their valuable comments and suggestions.

**Appendix**

**Proof of Theorem 1**

To prove this theorem, we need the following lemma.

**Lemma 1.** (i). For any two square matrices \( A \) and \( B \) with the same even number of rows \( A^* B^* = A^T B \).

(ii). For the \( S_c \) and \( T_c \) defined in (2), we have \( S_c^T S_c = S^*_c S^*_c = 2^c I_{2^c} \), \( S_c^T T_c + T_c^T S_c = (2^{2^c} + 2^c) I_{2^c} \), and 

\[
S_c^T T_c - T_c^T S_c = 0.
\]

**Proof of Lemma 1.** Conclusion (i) as well as the first equation in (ii) are obvious. We give only the proof of the second equation \( S_c^T T_c + T_c^T S_c = (2^{2^c} + 2^c) I_{2^c} \) in (ii); the proof of the third equation in (ii) is similar.

It is easy to see that equation \( S_c^T T_c + T_c^T S_c = (2^{2^c} + 2^c) I_{2^c} \) holds for \( c = 1 \). Suppose it holds for a particular integer \( c \) and consider the next integer \( c + 1 \). From (2), Lemma 1 (i) and the first equation in Lemma 1 (ii), we get

\[
S_{c+1}^T T_{c+1} = \begin{pmatrix}
2S^T_c T_c + 2^{2^c} I_{2^c} & -2^c S^*_c S^*_c \\
2^c S^*_c S^*_c & 2S^T_c T_c + 2^{2^c} I_{2^c}
\end{pmatrix}.
\]

Then

\[
S_{c+1}^T T_{c+1} + T_{c+1}^T S_{c+1} = (2(2^{2^c} + 2^c) + 2^{2^c+1}) I_{2^{c+1}} = (2^{2^c+2} + 2^{c+1}) I_{2^{c+1}},
\]

and the result follows by induction.

**Proof of Theorem 1.** (i) From the construction method, this assertion is obvious. (ii) First, we prove \( T_c^T T_c = t_c I_{2^c} \) by induction, where \( t_c = 2^c(2^{2^c} + 1)(2^{2^c+1} + 1)/6 \). It is easy to verify that \( T_1^T T_1 = 5I_2 \). Suppose \( T_c^T T_c = t_c I_{2^c} \), then we must show \( T_{c+1}^T T_{c+1} = t_{c+1} I_{2^{c+1}} \). In fact, by the induction hypothesis and Lemma 1, we have

\[
T_{c+1}^T T_{c+1} = \begin{pmatrix}
T_c^T & T_c^T + 2^c S_c^T \\
-(T_c^* + 2^c S_c^*) & T_c^* + 2^c S_c^*
\end{pmatrix}
\begin{pmatrix}
T_c & -(T_c^* + 2^c S_c^*) \\
T_c^* + 2^c S_c^* & T_c^*
\end{pmatrix}
\]

\[
= \begin{pmatrix}
(2t_c + 2^c(2^{2^c} + 2^c) + 2^{2^c}) I_{2^c} & 0 \\
0 & (2t_c + 2^c(2^{2^c} + 2^c) + 2^{2^c}) I_{2^c}
\end{pmatrix}
\]

\[
= t_{c+1} I_{2^{c+1}}.
\]

The fact that \( T_c^T T_c = t_c I_{2^c} \) proves that the \( L_c \) in (3) has property (a). So now we need only prove that \( L_c \) has property (b). For any three columns of \( L_c \), no matter whether they are distinct or not, they can be expressed as \( l_i = (l_{i1}, 0, -l_{i1}) \) with \( t_i \) being the corresponding column in \( T_c \) for \( i = 1, 2, 3 \). We then have

\[
(l_1 \odot l_2 \odot l_3)^T 1_{2^{c+1}} = (t_1 \odot t_2 \odot t_3)^T 1_{2^c} - (t_1 \odot t_2 \odot t_3)^T 1_{2^c} = 0,
\]

where \( a \odot b \) represents the elementwise product of \( a \) and \( b \), and \( 1_h \) denotes the \( h \times 1 \) column vector with all elements unity. Thus, property (b) of \( L_c \) follows immediately, and we complete the proof. \( \square \)
REFERENCES


[Received December 2008. Revised May 2009]