UCLA UCLA Electronic Theses and Dissertations

Title Constructions and Applications of Space-Filling Designs

Permalink https://escholarship.org/uc/item/0982k3qz

Author XIAO, QIAN

Publication Date 2017

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Constructions and Applications of Space-Filling Designs

A dissertation submitted in partial satisfaction of the requirements for the degree Doctor of Philosophy in Statistics

by

Qian Xiao

© Copyright by Qian Xiao 2017

ABSTRACT OF THE DISSERTATION

Constructions and Applications of Space-Filling Designs

by

Qian Xiao Doctor of Philosophy in Statistics University of California, Los Angeles, 2017 Professor Hongquan Xu, Chair

Maximin distance designs as an important class of space-filling designs are widely used in computer experiments, yet their constructions are challenging. In this thesis, we develop an efficient procedure to generate maximin Latin hypercube designs, as well as maximin multilevel fractional factorial designs, from existing orthogonal or nearly orthogonal arrays via level permutation and expansion. We show that the distance distributions of the generated designs are closely connected with the distance distributions and generalized word-length patterns of the initial designs. Examples are presented to show that our method outperforms many current prevailing methods. In addition, based on number theory and finite fields, we propose three algebraic methods to construct maximin distance Latin squares, as special Latin hypercube designs. We develop lower bounds on their minimum distances. The resulting Latin squares and related Latin hypercube designs have larger minimum distances than existing ones, and are especially appealing for high-dimensional applications. We show an application of space-filling designs in a combinatorial drug experiment on lung cancer. We compare four types of designs: a 512-run 8-level full factorial design, 80-run random sub-designs, 27-run random sub-designs and a 27-run space-filling three-level sub-design under four types of models: Kriging models, neural networks, linear models and Hill-based nonlinear models. We find that it is the best to adopt space-filling designs fitting Kriging models.

The dissertation of Qian Xiao is approved.

Arash Ali Amini

Jingyi (Jessica) Li

Qing Zhou

Hongquan Xu, Committee Chair

University of California, Los Angeles

2017

TABLE OF CONTENTS

1	Intr	$\operatorname{roduction}$										
2	Cor	nstruction of Maximin Distance Designs via Level Permutation and										
Ez	xpan	sion	4									
	2.1	Introduction										
	2.2	Some Theoretical Results	6									
	2.3	Maximin Distance Level Expansion (MDLE) Method	10									
		2.3.1 Procedures of MDLE	10									
		2.3.2 Justifications for the Procedures of MDLE	12									
	2.4	Results and Comparisons	14									
		2.4.1 Construction of Maximin LHDs	14									
		2.4.2 Construction of Maximin Fractional Factorial Designs	16									
	2.5	Multi-phase MDLE Method	19									
	2.6	Summary	21									
	2.7	Appendix: Proofs	21									
3	Con	nstruction of Maximin Distance Latin Squares and Related Latin Hy-										
pe	ercub	e Designs	26									
	3.1	Introduction	26									
	3.2	Costas Arrays and Welch Method	27									
	3.3	Gilbert Method	31									
	3.4	Golomb Method										
	3.5	Results and Comparisons	34									
	3.6	Appendix	37									

	3.7	Supple	ementary Material	40
		3.7.1	Additional Results and Comparisons	40
		3.7.2	Additional Proofs	41
4			n of Space-Filling Designs in a Drug Combinatorial Experiment	
on	Lun	ig Can	cer	44
	4.1	Introd	uction	44
	4.2	Constr	ruction of space-filling three-level designs	46
	4.3	Respon	nse surface modeling	46
		4.3.1	Kriging models	46
		4.3.2	Neural networks	48
		4.3.3	Linear and non-linear models	49
	4.4	Result	s and analysis	50
	4.5	Conclu	$sion \ldots \ldots$	60
5	Con	clusior	1	61

LIST OF FIGURES

2.1	Comparison of 9-run 2-factor LHDs	5
3.1	Difference triangles and cyclic Latin squares from Costas arrays	28
3.2	Ratio percentages for $(p-1) \times (p-1)$ Latin squares generated by Gilbert	
	method (solid), Golomb method (dashed), simplified Gilbert method (dotted),	
	and simplified Golomb method (dot-dash)	35
4.1	Examples of spatial correlation functions of Matérn family	48
4.2	Scatter-plots of predicted versus observed ATP levels on normal cells using	
	design MmD_{27}	52
4.3	Scatter-plots of predicted versus observed ATP levels on cancer cells using	
	design MmD_{27}	53
4.4	Comparison of predicted ATP levels via D_{full} and MmD_{27} under Kriging	
	models	54
4.5	Contour plots of predicted ATP levels via D_{full} and MmD_{27} under Kriging	
	models on normal cells	55
4.6	Contour plots of predicted ATP levels via D_{full} and MmD_{27} under Kriging	
	models on cancer cells.	56

LIST OF TABLES

2.1	Summary of minimum pairwise L_1 -distances $\ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots$	10
2.2	Comparisons of constructions of maximin LHDs	15
2.3	Comparisons in the constructions of four-level maximin FFDs \ldots .	17
2.4	Comparison in the construction of four-level and six-level uniform designs	18
2.5	Comparison of one-phase and two-phase MDLE methods in constructing LHDs	20
3.1	Comparison of L_1 -distance for $p \times (p-1)$ Latin hypercube designs	36
3.2	Comparison of L_1 -distance for $p \times p$ Latin hypercube designs	41
3.3	Comparison of L_1 -distance for $n \times n$ Latin hypercube designs with $n = p - 2$	42
4.1	Dose levels for each drug in the combinatorial experiment on lung cancer	45
4.2	Comparison of models and designs in fitting normal cell data	50
4.3	Comparison of models and designs in fitting cancer cell data	50
4.4	Estimations of parameters in Kriging models	57
4.5	Estimations of parameters in Kriging models for normal cells	58
4.6	Estimations of parameters in Kriging models for cancer cells	59

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincerest gratitude to my advisor, Prof. Hongquan Xu, for leading me into the door of research and guiding me along the way of my PhD period. He taught me not only knowledge on statistics, but also the way to think as a statistician. I'm very grateful that Professor Xu spent a lot of time revising papers with me and was very patient when I didn't have new ideas. Besides, I am grateful to all my committee members, Professors Arash Ali Amini, Jingyi (Jessica) Li and Qing Zhou, for serving on my committee. Their enlightening discussions and advice are very helpful.

I am very fortunate to work with Lin Wang, Jianfeng Yang, Fasheng Sun and Yaping Wang. Thanks for their help and advice on research and life.

I am very grateful to my parents and my wife Yang. All of this would not have been possible without their undivided love, support and encouragement. I would also like to thank my friends (alphabetical order) Chenhui Wang, Fan Zhang, Hanlu Su, Hao Ho, Jianwen Xie, Jianyu Wang, Jiaying Gu and Muzhou Liang.

This research was supported in part by National Science Foundation DMS Grant 1407560.

VITA

2012	B.A. (Investment Management), Central University of Finance and Eco-
	nomics, Beijing, China
2012-2013	Reader, Statistics Department, UCLA.
2013	Summer Session Lecturer, Department of Electrical Engineering, UCLA.
2013 - 2014	Teaching Assistant, Statistics Department, UCLA.
2015	M.S. (Statistics), UCLA.
2015 - 2017	Research Assistant and Teaching Assistant, Statistics Department, UCLA.
2017	Lecturer, Statistics Department, UCLA.

PUBLICATIONS

- Qian Xiao and Hongquan Xu (2017a). Construction of Maximin Distance Latin Squares and Related Latin Hypercube Designs. *Biometrika*, 104, 455-464.
- Qian Xiao and Hongquan Xu (2017b). Construction of Maximin Distance Designs via Level Permutation and Expansion. *Statistica Sinica*. in press.
- Lin Wang, Qian Xiao, and Hongquan Xu (2017). Optimal space-filling Latin hypercube designs based on good lattice point designs. Submitted.

CHAPTER 1

Introduction

The study of experimental design is extremely important in modern industry, science, and engineering. A good experimental design should minimize the number of runs needed to acquire as much information as possible. Much scientific research involves modeling complicated physical phenomena via complex computer codes. Scientists and engineers make use of computer experiments to explore the complex relationships between inputs and outputs, and to find an approximate model that is much simpler than the true but complicated model. In doing so, it is crucial to use a good space-filling design (Fang et al., 2006; Morris and Moore, 2015; Lin and Tang, 2015). The goal of space-filling designs is to bound the bias between the approximate model and the true model. There are two schools of thought on how to bound the bias. One approach is to spread the design points out as far from each other as possible consistent with staying inside the experimental boundaries. The other approach is to space the points out evenly over the region of interest. These designs are called space-filling designs. Space-filling designs include Latin hypercube designs (LHDs) and their modifications, maximin distance designs (Johnson et al., 1990) and uniform designs (Fang et al., 1999). A LHD is an $n \times k$ matrix of which each column is a permutation of levels $1, 2, \ldots, n$ or $0, 1, \ldots, n-1$. LHDs have uniform one-dimensional projections and orthogonal-array based LHDs (Tang, 1993) have improved two- or three-dimensional projections. Since computer experiments are deterministic and replicates should be avoided, these designs are desirable.

A maximin distance design spreads design points over the design space in such a way that the separation distance, i.e., the minimal distance between pairs of points, is maximized. Maximin distance designs are asymptotically *D*-optimal when observations are weakly correlated from a Gaussian process. The main idea of the uniform design is to uniformly scatter design points over the experimental domain, and Hickernell (1998) defined several discrepancy, including centered l_2 -discrepancy, to measure the uniformity of the design. Stochastic algorithms such as simulated annealing (Morris and Mitchell, 1995; Joseph and Hung, 2008; Ba et al., 2015), the threshold-accepting method (Winker and Fang, 1997; Fang et al., 2006) and swap or swarm optimization algorithms (Moon et al., 2011; Chen et al., 2013) have been used to construct maximin distance designs and uniform designs. However, such methods are not efficient for large designs due to the computational complexity. Nevertheless, large designs are needed for computer experiments; for example, Morris (1991) considered many simulation models involving hundreds of factors.

Space-filling fractional factorial designs are also widely used in physical experiments, e.g. drug combination experiments. Combinatory drugs have been broadly applied to treat various diseases since they often have higher efficacy and lower toxicity compared to individual drugs. It is challenging to quantify drug contributions and drug interactions for multiple drugs due to the inherent complexity of underlying biological systems (Al-Shyoukh et al., 2011). Various models should be used to analyze drug combination experiments, and for such situations space-filling designs are the most desirable (Zhou and Xu, 2014). However, few good designs and results are available in the literature because it is computationally expensive to construct designs when both the numbers of runs and factors are large.

In this thesis, we develop new methodology and design theory for space-filling designs. In Chapter 2, we propose an efficient procedure to generate maximin Latin hypercube designs, as well as maximin multilevel fractional factorial designs, from existing orthogonal or nearly orthogonal arrays via level permutation and expansion. We show that the distance distributions of the generated designs are closely connected with the distance distributions and generalized word-length patterns of the initial designs. Examples are presented to show that our method outperforms many current prevailing methods.

In Chapter 3, based on number theory and finite fields, we propose three algebraic methods to construct maximin distance Latin squares, as special Latin hypercube designs. We develop lower bounds on their minimum distances. The resulting Latin squares and related Latin hypercube designs have larger minimum distances than existing ones, and are especially appealing for high-dimensional applications. These algebraic constructions do not need any searching and thus can be used to generate large space-filling designs.

In Chapter 4, we present an application of space-filling designs in drug combination experiments on lung cancer. We focus on a 512-run combinatorial drug experiment by Al-Shyoukh et al. (2011). We compare four types of designs: a 512-run 8-level full factorial design, 80-run random sub-designs, 27-run random sub-designs and a 27-run three-level space-filling design under four major models: Kriging models, neural networks, linear models and Hill-based nonlinear models. We find that it is the best to adopt space-filling designs fitting Kriging models.

Chapter 2 is accepted for publication as "Construction of maximin distance designs via level permutation and expansion," *Statistica Sinica* in 2017. Chapter 3 is published as "Construction of maximin distance Latin squares and related Latin hypercube designs," *Biometrika* in 2017.

CHAPTER 2

Construction of Maximin Distance Designs via Level Permutation and Expansion

2.1 Introduction

Computer experiments are widely used in scientific researches and product developments to simulate real-world problems with complex computer codes (Santner et al., 2013; Fang et al., 2006; Morris and Moore, 2015). The most suitable designs for computer experiments are space-filling Latin hypercube designs (LHDs), yet their construction is challenging, especially for those with a large number of runs and factors.

Many researchers have studied orthogonal LHDs; see, among others, Steinberg and Lin (2006), Cioppa and Lucas (2007), Lin et al. (2009), Sun et al. (2010) and Yang and Liu (2012). However, orthogonal LHDs are not necessarily space-filling, e.g. design (a) in Figure 2.1. Another approach is through computer search using some optimality criteria based on discrepancy or distance. Hickernell (1998) defined several discrepancy criteria, and among them the centered L_2 -discrepancy (CD) is the most widely accepted. Johnson, Moore and Ylvisaker (1990) proposed the maximin and minimax distance criteria. In this chapter, we adopt the maximin distance criterion which seeks to scatter design points over the experimental domain such that the minimum distance between points is maximized. Johnson et al. (1990) showed that maximin distance designs are asymptotically optimal under a Bayesian setting. Morris and Mitchell (1995) proposed the criterion

$$\phi_p = \left(\sum_{i=2}^n \sum_{j=1}^{i-1} \frac{1}{d_{i,j}^p}\right)^{\frac{1}{p}},\tag{2.1}$$

where $d_{i,j}$ is the distance between the i^{th} and j^{th} row of the design. When p is sufficiently

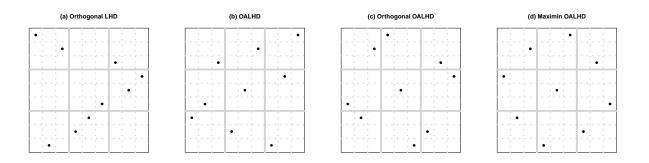


Figure 2.1: Comparison of 9-run 2-factor LHDs

large, ϕ_p is asymptotically identical to the maximin distance criterion. Morris and Mitchell (1995), Joseph and Hung (2008), Ba et al. (2015), and many others proposed algorithms to construct maximin LHDs; see Lin and Tang (2015) for a summary. To the best of our knowledge, the R package SLHD by Ba et al. (2015) is currently the most efficient algorithm.

Tang (1993) proposed to generate orthogonal array-based LHDs (OALHDs) by expanding levels in randomized orthogonal arrays (OAs). Though these OALHDs have desirable sampling and projection properties, most of them are not space-filling, e.g. designs (b) and (c) in Figure 2.1. A searching scheme can be applied to OALHDs (Leary et al. (2003)), but the results are not satisfactory. Ba et al. (2015) used a level expansion procedure similar to that of Tang (1993) when generating SLHDs with multiple slices. They justified their method from a geometric perspective but did not provide theoretical support. We provide some theoretical results to complement the work of Tang (1993) and Ba et al. (2015). We show that OAs, or nearly OAs if OAs do not exist, are good initial designs as they tend to generate robust space-filling designs. To avoid searching over the entire space of OALHDs generated via level expansion, we propose to perform level permutations on the initial designs and restrict level expansions only to the maximin OAs. Tang et al. (2012), Tang and Xu (2013), and Zhou and Xu (2014) used the level permutation method for constructing uniform and maximin fractional factorial designs, but their method cannot be used to construct LHDs and relies on the existence of multi-level OAs. We propose a procedure, the maximin distance level expansion (MDLE) method, to construct maximin designs by combining the strength of level permutation and expansion while avoiding their weaknesses. Our procedure is efficient, providing better designs using less time compared with existing methods. It is general, not only in the capability of constructing both maximin fractional factorial designs and maximin LHDs, but also in the flexibility to use multiple phases in level expansion that can significantly reduce the computation needed.

This chapter is organized as follows. We present our theoretical results in Section 2.2. In Section 2.3, we introduce the procedure, searching algorithm, and justifications for our MDLE method. In Section 2.4, examples are given to show that our method outperforms the ordinary level expansion method, the OMLHD method, the R package SLHD, and the level permutation method. In Section 2.5, we introduce a multi-phase method for constructing large maximin designs. Section 2.6 concludes, and all proofs are given in the Appendix.

2.2 Some Theoretical Results

Let $D(n, s^k)$ be an *n*-run, *k*-factor, and *s*-level (labelled as 1, 2, ..., s) balanced design where each level appears exactly n/s times in every column. From the initial design $D(n, s^k)$ we can generate a set of designs $D'(n, (ms)^k)$ with ms levels by a level expansion procedure. For each column in the initial design D, we replace the n/s positions of entry l (l = 1, 2, ..., s) by a random sequence of n/(ms) replicates of numbers: (l-1)m+1, (l-1)m+2, ..., (l-1)m+m, where n, k, s, m are all integers larger than 1 and n is divisible by ms. When m = n/s, the generated D's are LHDs.

Example 1. As an illustration, we perform the level expansion procedure to generate a $D'(8, 4^2)$ from a $D(8, 2^2)$. For each column in D, we first replace all four entries of 1 with a random permutation of numbers: 1,1,2,2, and then replace all four entries of 2 with a random permutation of numbers: 3,3,4,4, thus generating a 4-level design D'. In all we have 1296 possible D's. Here is an example:

$$D = \begin{pmatrix} 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 \\ 1 & 2 & 1 & 2 & 1 & 2 & 1 & 2 \end{pmatrix}^{T} \Rightarrow D' = \begin{pmatrix} 1 & 2 & 1 & 2 & 3 & 4 & 4 & 3 \\ 2 & 3 & 1 & 3 & 1 & 4 & 2 & 4 \end{pmatrix}^{T}$$

Let $x_{i,l}$ be the (i^{th}, l^{th}) element and x_i be the i^{th} row of the initial design D. After level expansion, they are $x'_{i,l}$ and x'_i of the generated design D', respectively. Let $h_{i,j}$ be the Hamming distance (number of positions where the corresponding entries in the pair of rows are different) between rows x_i and x_j . Take $d_{il,jl} = |x_{i,l} - x_{j,l}|$. Denote the L_1 -distance between two rows x_i and x_j as $d_{i,j} = \sum_{l=1}^k d_{il,jl}$. In this paper, we focus on constructing maximin designs in regard to the L_1 -distance. Let $d_{min}(D)$ be the minimum L_1 -distance among all pairs of rows in design D. In the same way, we define $h'_{i,j}$, $d'_{i,jl}$, $d'_{i,j}$ and $d_{min}(D')$ for the generated design D', respectively. For any balanced design D, we define the distance distribution as (# denotes the count)

$$B_l(D) = n^{-1} \# \{ (i, j) : d_{i,j} = l; x_i, x_j \in D, i, j = 1, 2, \dots, n \}.$$

It is easy to show that a design without repeated runs has $B_0(D) = 1$. The maximin design is defined as the one that sequentially minimizes the distance distribution $B_0(D), B_1(D),$ $B_2(D), B_3(D), \ldots$ Designs with smaller ϕ_p values defined in (2.1) are more space-filling and have better distance distributions.

Lemma 1. (a) For i, j = 1, ..., n and $i \neq j$, upper and lower bounds for the L_1 -distance between the i^{th} and j^{th} row in the generated design D' are

$$md_{i,j} - (m-1)h_{i,j} \le d'_{i,j} \le md_{i,j} + (m-1)k.$$

(b) Upper and lower bounds for the minimum pairwise L_1 -distance of the generated design D' are

$$md_{min}(D) - (m-1)h_{max}(D) \le d_{min}(D') \le md_{min}(D) + (m-1)k_{max}(D)$$

where $h_{max}(D)$ is the largest pairwise Hamming distance in design D.

Given n, s, and k from different initial designs $D(n, s^k)$, by level expansion we can generate different sets of designs $D'(n, (ms)^k)$. By Lemma 1, the upper bound for $d_{min}(D')$ is determined by $d_{min}(D)$. If we can generate a design D'_{opt} with $d_{min}(D'_{opt}) = md_{min}(D_{Mm}) + (m-1)k$ where D_{Mm} is the maximin initial design, it is clear that D'_{opt} has the largest minimum distance among all possible D's from all possible initial designs D. In Lemma 1, the lower bound of $d_{min}(D')$ is also positively related with $d_{min}(D)$. Therefore, in order to generate good maximin designs via level expansion, initial designs with better distance distributions should be used.

From any initial design D, by level expansion we have $((n/s)!/(r!)^m)^{sk}$ possible generated designs D', where r = n/(ms).

Theorem 1. For i, j = 1, ..., n and $i \neq j$, the expectation and variance of the pairwise L_1 -distances in the generated designs D' via level expansion have the following relationship with the pairwise L_1 -distance in the initial balanced design D:

$$E(d'_{i,j}) = md_{i,j} + (k - h_{i,j})\gamma$$
 and $Var(d'_{i,j}) = C_{1,0} + C_{1,1}h_{i,j}$,

where $\gamma = n(m^2 - 1)/[3m(n - s)], C_{1,0} = kn(m^2 - 1)(m^2n + 2n - 3m^2s)/[18m^2(n - s)^2],$ and $C_{1,1} = (m^2 - 1)[2n^2(m^2 - 1) - 3m^2s(n - s)]/[18m^2(n - s)^2].$

Thus the expected value of $d'_{i,j}$ is a function of both $d_{i,j}$ and $h_{i,j}$. For a 2-level design, the L_1 -distance $d_{i,j}$ equals the Hamming distance $h_{i,j}$. For a design with more than 2 levels, $d_{i,j}$ is greater than or equal to $h_{i,j}$. In addition, the coefficient (m) for $d_{i,j}$ is almost three times as large as the absolute value of the coefficient (γ) for $h_{i,j}$. Therefore, the expected value of $d'_{i,j}$ is dominated by $d_{i,j}$. Generally speaking, a large $d_{i,j}$ value leads to a large $d'_{i,j}$ value on average.

When s > 2, we can improve designs' minimum distances by level permutation (Zhou and Xu (2014)). When permuting levels for one or more factors of a design, the pairwise Hamming distances do not change, but its pairwise L_1 -distances vary. Given a design $D(n, s^k)$, we can generate in total $(s!)^k$ level-permuted designs (including isomorphic designs) and then consider all possible level expansions for each design. Let Θ denote the set of all designs generated by all level permutations and expansions.

Lemma 2. When all possible level permutations and expansions are considered, for $i \neq j$, the expectation and variance of the pairwise L_1 -distances in generated designs D' are

$$E_{\Theta}(d'_{i,j}) = k\gamma + (m\frac{s+1}{3} - \gamma)h_{i,j}$$
$$Var_{\Theta}(d'_{i,j}) = C_{1,0} + \left(C_{1,1} + m^2\frac{(s+1)(s-2)}{18}\right)h_{i,j},$$

where γ , $C_{1,0}$, and $C_{1,1}$ are constants defined in Theorem 1.

Now we study the space-filling property for the generated design D'. For $D' \in \Theta$, let $\overline{d'} = \sum_{i \neq j=1}^{n} d'_{i,j}/(n(n-1))$ be the average distance in the generated design D'. It is easy to show that $\overline{d'} = kn(m^2s^2 - 1)/(3ms(n-1))$ because D' is level balanced for each column. Next, we show that the expectation of sum of squared distances in D' is minimized when the initial design is an OA. The concepts of generalized word-length pattern (GWLP) and generalized minimum aberration (GMA) from Xu and Wu (2001) are needed to describe this result. For design $D(n, s^k)$, the GWLP is the vector $(A_1(D), A_2(D), \ldots, A_k(D))$, where the value of $A_j(D)$ (j = 1, 2..., k) represents the total aliasing between the general mean and all j-factor interactions in the full ANOVA model. The GMA criterion sequentially minimizes the GWLP.

Theorem 2. When all possible level permutations and expansions are considered,

$$E_{\Theta}(\sum_{i\neq j=1}^{n} (d'_{i,j})^2) = C_{2,1}A_2(D) + C_{2,0},$$

where $C_{2,1} = 2n^2(m(s+1)/3 - \gamma)^2/s^2$ and $C_{2,0}$ is a constant.

From Theorem 2, we have $E_{\Theta}(\sum_{i\neq j=1}^{n} (d'_{i,j} - \bar{d}')^2) = E_{\Theta}(\sum_{i\neq j=1}^{n} (d'_{i,j})^2) - E_{\Theta}(\sum_{i\neq j=1}^{n} (\bar{d}')^2)$ = $C_{2,1}A_2(D) + constant$. Since $C_{2,1} > 0$, the expectation of the variation of pairwise L_1 distances in D' is minimized when $A_2(D) = 0$. For a level balanced design, $A_1(D) = 0$. Xu and Wu (2001) showed that D is an OA of strength two if and only if $A_1(D) = A_2(D) = 0$. Thus, if the initial design is an OA of strength two or higher, generated designs tend to have small variations among all pairwise L_1 -distances and large minimum pairwise L_1 -distance. In other words, designs generated from OAs via level permutation and expansion tend to have robust space-filling properties.

Example 2. Consider constructing 32-run LHDs with 8 factors from five different 2-level designs with different A_2 or A_3 values. The first two designs are regular 2^{8-3} designs (with $A_2 = 0$) and the other three designs have 1, 2, 3 pairs of duplicated columns, indicated by $A_2 = 1$, 2, 3, respectively. Given a 2-level design, we randomly generated 10^5 LHDs via

Design	(A_1, A_2, A_3, A_4)	Min	Q_1	Median	Q_3	Max
Design 1	(0,0,0,3)	15	36	39	42	52
Design 2	(0, 0, 0, 0, 0) (0, 0, 1, 2)	15	35	38	41	51
0						0 -
Design 3	(0, 1, 0, 2)	14	33	36	39	49
Design 4	(0, 2, 0, 1)	11	32	35	38	48
Design 5	(0,3,0,3)	10	29	32	34	45

Table 2.1: Summary of minimum pairwise L_1 -distances

level permutation and expansion and computed the minimum pairwise L_1 -distance for each of them. Table 2.1 compares the minimum, first quartile (Q1), median, third quartile (Q3) and maximum of the 10⁵ minimum distances for five different initial designs. It is evident that initial designs with smaller A_2 values are more likely to generate designs with larger minimum distances via level permutation and expansion.

It is possible, but tedious, to extend Theorem 2 and link $E_{\Theta}(\sum_{i\neq j=1}^{n} (d'_{i,j})^r)$ with the values of $A_2(D), \ldots, A_r(D)$ for r > 2, similar to Theorem 4 of Zhou and Xu (2014). We do not pursue this here.

2.3 Maximin Distance Level Expansion (MDLE) Method

2.3.1 Procedures of MDLE

Based on the results in the previous section, we propose the MDLE method that combines both level permutation and expansion. The method starts from OAs, or nearly-OAs if the corresponding OAs are not available, and expands their levels with one or more phases. Here we first discuss how to construct maximin designs from OAs with only one phase of level expansion. Refer to Section 2.5 for generalizations. To generate $D'(n, (ms)^k)$, we start from an $OA(n, s^{k_0})$ with $k_0 \ge k$. The MDLE method has three steps.

1. Select the GMA k-column subset from an $OA(n, s^{k_0})$ and denote it by $D(n, s^k)$.

- 2. If s > 2, perform level permutation for design D from Step 1. Select the maximin design and denote it by $D_p(n, s^k)$.
- 3. For each column in D_p from Step 2, replace the n/s positions of entry l (l = 1, 2, ..., s) by a random sequence of n/(ms) replicates of numbers: (l−1)m+1, (l−1)m+2..., (l−1)m+m. Select the maximin design as the final design D'(n, (ms)^k).

We usually start from saturated $OA(n, s^{k_0})$, or nearly saturated OAs with $k_0 \leq (n-1)/(s-1)$. When $k_0!/(k!(k_0-k)!)$ is small, we can enumerate and compare all subsets to find the GMA subset in Step 1; otherwise, we adopt a simple searching method: randomly generate and compare n_{gma} subsets and select the GMA subset where n_{gma} ranges from 1000 to 5000 based on the design size and computation available. We use the concept of minimum moment aberration (Xu (2003)) to efficiently determine GMA subsets. For 2-level regular designs we choose existing minimum aberration designs from the R package FrF2. In Steps 2 and 3, we adopt a threshold accepting (TA) algorithm modified from that of Dueck and Scheuer (1990). Compared with the simulated annealing algorithm by Kirkpatrick (1984) and Morris and Mitchell (1995), TA converges faster.

To implement the TA algorithm, we need to specify neighbour designs $\mathcal{N}(D_c)$ for a current design D_c in Steps 2 and 3. To generate neighbour designs $\mathcal{N}(D_c)$ in Step 2, we randomly choose two levels from a randomly chosen column of D_c and exchange all elements of these two levels. In Step 3, we define neighbour designs $\mathcal{N}(D_c)$ by exchanging the levels in two positions from a randomly chosen column of D_c , where these two positions have different values in D_c and the same value in D_p from Step 2.

We choose $\phi(D) = \phi_p(D)$ defined in (2.1) as the objective function to be minimized in our TA algorithm. The pseudo code for our TA algorithm is given in Algorithm 1. Based on the design size and time limits, typically we set n_{seq} equal to 2000, choose n_{rounds} from 30 to 75, and choose n_{steps} from 3000 to 7500.

Algorithm 1 Pseudo code for threshold accepting (TA) algorithm

Initialize n_{seq} (number of steps to compute threshold sequences) Initialize n_{rounds} (number of rounds) and n_{steps} (number of steps) Initialize a starting design D_c and let $D_{min} = D_c$ for i = 1 to n_{seq} do Generate a new design D_n from its neighbors $\mathcal{N}(D_c)$ and let $\Delta_i = |\phi(D_c) - \phi(D_n)|$ end for Compute the empirical distribution of Δ_i , $i = 1, 2, \ldots, n_{seq}$, denoted it as F(x)for r = 1 to n_{rounds} do Generate threshold $\tau_r = F^{-1} \left(0.5 (1 - r/n_{rounds}) \right)$ for j = 1 to n_{steps} do Generate a new design D_n from the neighbors $\mathcal{N}(D_c)$ and let $\delta = \phi(D_n) - \phi(D_c)$ if $\delta < \tau_r$ then let $D_c = D_n$ if $\phi(D_c) < \phi(D_{min})$ then let $D_{min} = D_c$ end for end for Return D_{min}

2.3.2 Justifications for the Procedures of MDLE

Zhou and Xu (2014) showed that from GMA initial designs we can generate designs with the best distance distributions on average via all possible level permutations. Thus, choosing GMA design D in Step 1 can benefit finding maximin design D_p in Step 2. Further, Lemma 1 and Theorem 1 in Section 2.2 show that from the maximin design D_p we can generate D's with the best distance distributions on average in Step 3. By Theorem 2, GMA initial designs minimize the expectation of the variation of distances in generated designs via level permutation and expansion. Therefore, this 3-step procedure is robust and efficient in generating good space-filling designs.

We further justify our method from a geometric point of view. Ba et al. (2015) discussed a relevant geometric idea, but it only applies to SLHDs with multiple slices. Here we discuss the situations for fractional factorial designs and general LHDs, including SLHDs with only one slice. We relate a design's geometric structure with its GMA structure. To get a space-filling *n*-run and *k*-factor design, a straightforward idea is to divide the design space equally into *n k*-dimensional lattices, put one point in each lattice, and properly adjust each point's position within its lattice. This geometric structure of "one point per lattice" can be achieved by performing level expansion to full factorial initial designs. For example, see OALHDs in Figure 2.1(b),(c), and (d) generated by the level expansion process from full factorial $D(9, 3^2)$. These designs have only one point per lattice formed by the solid lines, but the positions of points within the lattices are different. By either the level permutation or level expansion process, the "one point per lattice" structure is not changed, but their positions within the lattice are adjusted, and thus the distance distribution of the design can be improved. By our MDLE method with full factorial initials, we can find the design with best distance distribution while keeping the "one point per lattice" structure, e.g., design (d) in Figure 2.1.

As a generalization, when $n < s^k$, an initial design with the most low-dimensional projections that are full factorials is ideal for our MDLE method, and GMA designs have such a property in many cases. Box and Hunter (1961) pointed out that any *p*-dimensional (p < r) projection of a 2-level regular design with resolution r is a full-factorial design. Chen (1998) proved that for a 2-level regular design, $\binom{n}{p} - \sum_{j=r}^{p} \binom{n-j}{p-j} A_j(D)$ *p*-dimensional projections $(p = r, \ldots, \lfloor r + (r-1)/2) \rfloor$) are full-factorial designs. Under these cases, since the GMA initials have largest resolutions and sequentially minimize $A_j(D)$ ($j = 1, 2, \ldots k$), they have the most parts that are full-factorials in *p*-dimensional projection spaces ($p \leq \lfloor r + (r-1)/2 \rfloor \rfloor$). GMA nonregular designs have similar projection properties; see Xu et al. (2009) for a review. As a result, GMA initial designs tend to generate better space-filling designs via level expansion.

2.4 Results and Comparisons

2.4.1 Construction of Maximin LHDs

First, we compared our MDLE method with the ordinary level expansion (OLE) method of Tang (1993) and Leary et al. (2003) in generating maximin OALHDs. The OLE method first randomly selects a required number of columns from a saturated or nearly saturated OA to be the initial design, then performs level expansion, and searches for the maximin generated LHDs. In order to make a fair comparison, we replaced the simulated annealing algorithm in Leary et al. (2003) with our more efficient TA algorithm.

Table 2.2 lists some arbitrarily chosen cases for comparison, where "d(pair)" represents the minimum pairwise L_1 -distance (and the number of pairs with the minimum distance). For all tables, we use bold font to represent the better results. For the 32, 64 and 128run cases, the MDLE method starts from the respective 2-level minimum aberration initial designs that are available in R package FrF2, whereas the OLE method starts from the corresponding saturated OAs. For the 27, 54, 81, and 125-run cases, both methods start from initial designs $OA(27, 3^{13})$, $OA(54, 3^{25})$, $OA(81, 3^{40})$, and $OA(125, 5^{31})$, respectively; these are available in R package DoE.base. All codes were run in R on a laptop with an Intel 2.50GHz I7 CPU. Time used by our MDLE method ranged from 5 minutes to an hour for the different cases here. For all cases, we let the OLE method use at least twice as much time as the MDLE method.

From Table 2.2, it is clear that the MDLE method generates better OALHDs than the OLE method for all cases. Compared with the MDLE method, the OLE method only includes Step 3, but does not have the first two steps of the MDLE method. Thus, Table 2.2 shows the usefulness of the first two steps in the MDLE method which provides good initial designs for level expansion. When the MDLE method starts with 2-level initial designs, Step 2 is skipped since level permutations do not change designs' distance distributions. Thus, the usefulness of Step 1 alone can be seen from the 32, 64 and 128-run cases in Table 2.2. From the 54-run/25-factor, 81-run/40-factor and 125-run/31-factor cases, we can see the usefulness of Step 2 alone since Step 1 is skipped.

			MDLE		OLE	OMI	THD	SLH	D
n	k	d(pair)	ψ_p	d_2	d(pair)	d(pair)	ψ_p	d(pair)	d_2
27	9	72(2)	0.012	28.8	68(5)	60(1)	0.025	63(1)	28.2
32	20	205(1)	0.005	55.6	205(2)	177(1)	0.012	190(1)	55.1
54	5	54(1)	0.0311	28.1	45(1)	47(2)	0.0393	44(1)	27.8
54	20	329(1)	0.0034	88.4	317(1)	279(1)	0.0083	294(1)	88.2
54	25	425(3)	0.0022	102.7	399(1)	360(1)	0.012	382(1)	100.9
64	6	83(1)	0.0209	40.6	61(1)	70(3)	0.0299	67(1)	39.1
64	20	378(1)	0.0034	105.2	369(1)	310(1)	0.0084	340(1)	102.6
64	40	813(1)	0.0025	157.4	804(1)	698(1)	0.0048	771(1)	155.7
81	8	152(1)	0.0111	64.2	102(1)	123(2)	0.0198	121(1)	62.7
81	25	604(2)	0.0022	147.9	577(1)	504(1)	0.0028	540(1)	146.7
81	40	1016(1)	0.0016	194.9	962(1)	899(1)	0.0016	934(1)	194.5
125	10	284(2)	0.0072	111.8	199(1)	237(3)	0.0136	232(1)	110.6
125	23	797(1)	0.0021	206.9	640(1)	668(1)	0.0021	726(1)	206.8
125	31	1126(1)	0.0014	251.1	971(1)	955(1)	0.0076	1038(1)	250.8
128	12	378(1)	0.0051	135.5	284(1)	314(1)	0.0092	313(1)	132.6
128	49	1893(1)	0.0014	337.6	1873(1)	1643(1)	0.0057	1801(1)	335.3
128	64	2512(1)	0.0017	395.2	2479(1)	2239(1)	0.0061	2497(1)	392.1

Table 2.2: Comparisons of constructions of maximin LHDs

Next, we compared our MDLE method with the OMLHD method of Joseph and Hung (2008) and the R package SLHD of Ba et al. (2015) in generating space-filling LHDs. Joseph and Hung (2008) proposed the multi-objective criterion

$$\psi_p = \omega \rho^2 + (1 - \omega) \frac{\phi_p - \phi_{p,lb}}{\phi_{p,ub} - \phi_{p,lb}},$$
(2.2)

where ϕ_p is defined in (2.1) with p = 15, ρ^2 is the average of squared column-wise correlations, ω is the weight which is set to 0.5, $\phi_{p,lb}$ and $\phi_{p,ub}$ are the smallest and largest possible ϕ_p values. Joseph and Hung (2008) used a modified simulated annealing algorithm to search for LHDs that minimize ψ_p values. Table 2.2 lists some cases for comparison, where ψ_p is defined in (2.2) and d_2 represents designs' minimum pairwise L_2 -distances. For the OMLHD method, we ran the code from Y. Hung's homepage (http://stat.rutgers.edu/home/yhung /index.htm) with nstart = 5 and default settings, and chose the best results. For the SLHD method, we ran the command maximinSLHD with slice parameter t = 1 and default settings for 200 times, and chose the best results.

For all cases in Table 2.2, the MDLE method generates better space-filling designs than the OMLHD method in regard to both the L_1 -distance and the ψ_p criterion. Our MDLE method searches designs toward the L_1 -distance alone. Designs from our method have small pairwise correlations, since they can always collapse to OAs or nearly OAs. Further, the MDLE method generates better maximin designs than the SLHD method under both the L_1 - and L_2 -distances. In order to make a fair comparison with the SLHD package, the ϕ_p criterion used in the MDLE method adopted the L_1 - and L_2 -distance for each case respectively. Our MDLE method was implemented in R whereas the SLHD and OMLHD methods were implemented in C++. The R package SLHD provides an interface to call the C++ program. Our MDLE method used less than half of the time used by the SLHD and OMLHD methods, although C++ is more efficient than R in terms of computation.

2.4.2 Construction of Maximin Fractional Factorial Designs

First, we compared our MDLE method with the level permutation (LP) method of Zhou and Xu (2014) in generating maximin fractional factorial designs (FFDs). Zhou and Xu (2014)

		MDLE	LP
n	k	d(pair)	d(pair)
16	3	2(12)	2(12)
16	4	4(60)	4(56)
16	5	4(1)	4(4)
32	3	2(156)	2(156)
32	4	2(8)	2(8)
32	5	4(100)	4(106)
32	6	5(48)	5(58)
32	7	6(24)	6(28)
32	8	8(132)	8(128)
32	9	9(62)	8(6)

Table 2.3: Comparisons in the constructions of four-level maximin FFDs

(a)

included a table of 10 maximin designs with $n \leq 32$ that are comparable here, and we list them in Table 2.3 (a). We further selected another eight larger cases with $n \geq 48$ in Table 2.3 (b) to compare the two methods. All designs are 4-level FFDs. For the MDLE method, in the 16, 32, 64, and 128-run cases, 2-level minimum aberration initial designs were used; in the 48 and 80-run cases, $OA(48, 2^{47})$ and $OA(80, 2^{79})$ were used as the initial designs. For the LP method, in the 48, 64, 80 and 128-run cases, initial designs $OA(48, 4^{13})$, $OA(64, 4^{11})$, $OA(80, 4^{11})$ and $OA(128, 4^{40})$ were used, respectively. Both methods' codes were run in R. For all cases, the LP method used at least twice as much time as the MDLE method.

In Table 2.3 (a), for the first nine cases both methods generated designs with the same minimum pairwise distances. For the last case in Table 2.3 (a) and all cases in Table 2.3 (b), the MDLE method outperforms the LP method. Furthermore, the LP method relies on existing OA initials that have the same number of runs, factors and levels as the generated designs. These OAs are often difficult to find or even do not exist. For example, there is no $OA(24, 6^8)$ that can be used to generate maximin $D'(24, 6^8)$. Compared with the LP

			MD	DLE	MDL	E-CD	UD-page designs	
n	k	s	CD	d(pair)	CD	d(pair)	CD	d(pair)
32	7	4	0.074	6(18)	0.070	5(6)	0.071	4(1)
32	13	4	0.0343	13(5)	0.0343	13(5)	0.0344	12(2)
40	13	4	0.3186	13(4)	0.3067	12(5)	0.3068	11(1)
40	15	4	0.5080	16(56)	0.4969	13(1)	0.4987	14(1)
48	11	4	0.1841	10(11)	0.1758	8(1)	0.1767	7(1)
48	15	4	0.461	15(13)	0.447	13(1)	0.449	12(1)
36	12	6	0.1744	20(15)	0.1673	19(14)	0.1691	17(1)
48	12	6	0.1416	19(21)	0.1362	16(2)	0.1374	16(1)
54	9	6	0.0601	12(13)	0.0564	8(1)	0.0568	10(3)
54	12	6	0.1362	17(14)	0.1268	16(1)	0.1299	16(2)
60	9	6	0.0576	12(11)	0.0544	8(1)	0.0546	9(2)

Table 2.4: Comparison in the construction of four-level and six-level uniform designs

method, our MDLE method has more flexibility in design size, since we can start from 2-level designs to generate multi-level designs. For example, we can start from a 2-level $OA(24, 2^{23})$ to generate the 24-run/6-level maximin design with up to 23 factors.

Next, we compared designs from our MDLE method with some existing uniform designs listed on the uniform design homepage (http://uic.edu.hk/isci/). These uniform designs (UD-page designs) were searched by Kaitai Fang and his collaborators toward the centered L_2 -discrepancy (CD) criterion where smaller CD values indicate more space-filling designs. In order to make a fair comparison, in Table 2.4 we also include a modified version of our MDLE method (MDLE-CD) which searches best designs using the CD criterion in Step 3. We selected some 4-level and 6-level cases for comparison in Table 2.4. Both the MDLE and MDLE-CD methods started from the initial designs $OA(32, 2^{31})$, $OA(40, 2^{39})$, and $OA(48, 2^{47})$ to generate the 4-level designs, and $OA(36, 3^{13})$, $OA(48, 2^{47})$, $OA(54, 3^{18})$, and $OA(60, 2^{30})$ to generate the 6-level designs for the corresponding cases.

Table 2.4 shows that designs by the MDLE method are always better than the UD-page

designs in regard to maximin distance criterion. Designs from the MDLE-CD method are better than the UD-page designs toward the CD criterion.

2.5 Multi-phase MDLE Method

In constructing maximin designs $D'(n, (ms)^k)$ from initial designs $D(n, s^k)$, when m is very large, the one-phase MDLE method introduced in Section 2.3 is not efficient because level expansion produces too many designs. In addition, when n and k are too large given the computation constraint, we need to further restrict the searching space in the MDLE method. Under such situations, we can apply a multi-phase MDLE method. The multi-phase MDLE method shares the same Steps 1 and 2 as the one-phase MDLE. The difference lies in Step 3: instead of directly generating $D'(n, (ms)^k)$ from $D_p(n, s^k)$, we gradually expand the levels from s to ms in multiple phases. For example, in a two-phase MDLE method with $m = m_1m_2$, in Step 3 we first generate maximin design $D_1(n, (m_1s)^k)$ via level expansion from D_p in Step 2; then from D_1 we generate maximin design $D_2(n, (m_2m_1s)^k)$ which is $D'(n, (ms)^k)$ via level expansion again. It is straightforward to generalize and justify this process with more phases in both theory and geometry, as in Section 2.3.2.

The more phases we use, the more restrictions are put on the searching space. The number of designs needed to be compared decreases dramatically with multiple phases. For example, to generate $D'(16, 8^2)$ from $D(16, 2^2)$ for the one-phase MDLE method, we have in total about 4×10^{13} possible D's to be compared; for the two-phase MDLE method, we only need to compare about 1.7×10^6 designs. More restrictions on the searching space also means that we are more likely to miss good designs, at least in theory. In practice, with limited computations, the multiple-phase method can be more efficient than the one-phase method, especially for large designs.

Table 2.5 compares the one-phase and two-phase MDLE methods in generating maximin LHDs with time constraints. For both methods, we started from the respective full factorial designs for the first five cases, the minimum aberration designs for the 32- and 64-run cases, $OA(54, 3^{24})$, $OA(81, 3^{40})$, and $OA(125, 5^{31})$ for the rest of cases, respectively. For the last

		One-pl	nase		Two-p	hase
n	k	d(pair)	time	d(pair)	time	sequence
27	3	14(4)	67	14(5)	107	$3 \rightarrow 9 \rightarrow 27$
32	5	37(1)	103	37(3)	101	$2 \rightarrow 8 \rightarrow 32$
64	6	83(1)	301	81(1)	306	$2 \rightarrow 8 \rightarrow 64$
81	4	50(1)	478	50(3)	490	$3 \rightarrow 9 \rightarrow 81$
125	3	38(5)	603	37(3)	950	$5 \rightarrow 25 \rightarrow 125$
32	15	151(1)	211	150(2)	218	$2 \rightarrow 8 \rightarrow 32$
54	12	173(1)	886	178(2)	806	$3 \rightarrow 9 \rightarrow 54$
54	20	309(1)	1346	322(2)	1275	$3 \rightarrow 9 \rightarrow 54$
64	40	805(1)	1062	810(1)	995	$2 \rightarrow 8 \rightarrow 64$
81	40	1005(1)	1479	1014(1)	936	$3 \rightarrow 9 \rightarrow 54$
125	31	1111(1)	2085	1116(1)	1548	$5 \rightarrow 25 \rightarrow 125$

Table 2.5: Comparison of one-phase and two-phase MDLE methods in constructing LHDs

Note: Time in seconds.

five cases, where the numbers of runs and factors are relatively large, the two-phase method generates better designs in a shorter time than the one-phase method. Given adequate computation time, the one-phase method eventually generates better designs than the twophase method; see the last four cases in Table 2.5 and corresponding results in Table 2.2 where we ran the one-phase MDLE method for a longer time.

When OAs with different levels exist, generally speaking, it is better to use OA initials with fewer levels given abundant computations. As an illustration, for the 128-run/12-factor case, starting from the 2-, 4- and 8-level OA initials, the one-phase MDLE method generates LHDs with the minimum L_1 -distances of 378, 375 and 368, respectively. The 2-level OA initial gives the best result here, but requires more than 5 times the computations to achieve a stable result compared with the 8-level initial. Since any 8-level OA can be collapsed to a 2-level OA, the MDLE method is less likely to miss good results from 2-level OAs. For large designs with computation constraints, OA initials with larger levels may work better since the searching space is much smaller and less phases are needed.

2.6 Summary

We propose the MDLE method which can efficiently generate maximin LHDs and maximin fractional factorial designs. To justify our method, we establish a relationship of the L_1 distance distributions between the initial and generated designs via level expansion. When all possible level permutations of the initial designs are considered, we give expectations and variances of the pairwise L_1 -distances for the generated designs. Various comparisons show that our MDLE method outperforms the ordinary level expansion process, the OMLHD algorithm, the SLHD package, and the level permutation method. We also find many more space-filling designs compared to the existing uniform designs.

The MDLE method is easy to generalize for mixed-level cases. Starting from a mixedlevel initial design, we can individually set the level expansion path for each factor. In this way, we can generate mixed-level factorial designs. Although the MDLE method cannot generate maximin designs with any run size, it works well from nearly OAs (Xu (2002)) or optimal supersaturated designs (Xu and Wu (2005)), when suitable OA initials are not available.

2.7 Appendix: Proofs

Proof of Lemma 1. (a) For $i \neq j$, when $x_{i,l} = x_{j,l}$, $d'_{il,jl}$ takes on values of $0, 1, \ldots, m-1$; when $x_{i,l} \neq x_{j,l}$, $d'_{il,jl}$ takes on values of $m(d_{il,jl}-1)+1, \ldots, m(d_{il,jl}-1)+2m-1$. Therefore, the smallest possible $d'_{i,j}$ value is

min
$$d'_{i,j} = 0 * (k - h_{i,j}) + \sum_{l=1}^{h_{i,j}} (m(d_{il,jl} - 1) + 1) = md_{i,j} - (m - 1)h_{i,j}$$

and the largest possible $d'_{i,j}$ value is

$$\max d'_{i,j} = (m-1)(k-h_{i,j}) + \sum_{l=1}^{h_{i,j}} (m(d_{il,jl}-1) + 2m-1) = md_{i,j} + k(m-1).$$

Thus, we have $md_{i,j} - (m-1)h_{i,j} \le d'_{i,j} \le md_{i,j} + k(m-1).$

(b) Let x_a and x_b be the pair of rows in design D that forms the minimum pairwise L_1 -distance $d_{min}(D)$ (there could be many such pairs). Let x'_c and x'_d be the pair of rows in design D' that forms the minimum pairwise L_1 -distance $d_{min}(D')$ (there could be many such pairs). Then

$$d_{min}(D') = d'_{c,d} \le d'_{a,b} \le md_{a,b} + k(m-1) = md_{min}(D) + (m-1)k,$$

$$d_{min}(D') = d'_{c,d} \ge md_{c,d} - (m-1)h_{c,d} \ge md_{c,d} - (m-1)h_{max}(D)$$

$$\ge md_{a,b} - (m-1)h_{max}(D) = md_{min}(D) - (m-1)h_{max}(D).$$

Thus, we have $md_{min}(D) - (m-1)h_{max}(D) \le d_{min}(D') \le md_{min}(D) + (m-1)k$.

Proof of Theorem 1. We first calculate the probability distribution for $d'_{il,jl}$ with its range given in Lemma 1. For $i \neq j$, when $x_{i,l} = x_{j,l}$, the probability distribution is

$$P(d'_{il,jl} = 0) = \frac{m\binom{n/(ms)}{2}}{m(m-1)(n/(ms))^2 + m\binom{n/(ms)}{2}} = \frac{n-ms}{m(n-s)}$$

$$P(d'_{il,jl} = t) = \frac{2(m-t)(n/(ms))^2}{m(m-1)(n/(ms))^2 + m\binom{n/(ms)}{2}} = \frac{2n(m-t)}{m^2(n-s)}$$

for t = 1, 2, ..., m - 1. Thus,

$$E(d'_{il,jl}) = \sum_{t=1}^{m-1} t P(d'_{il,jl} = t) = \frac{n(m^2 - 1)}{3m(n-s)} = \gamma,$$
(2.3)

$$E((d'_{il,jl})^{2}) = \sum_{t=1}^{m-1} t^{2} P(d'_{il,jl} = t) = \frac{n(m^{2} - 1)}{6(n-s)} = \frac{m}{2} \gamma.$$
(2.4)

When $x_{i,l} \neq x_{j,l}$, the probability distribution is

$$P(d'_{il,jl} = d_0 + t) = \frac{t+1}{m^2}, \text{ for } t = 0, 1, \dots, m-1,$$
$$P(d'_{il,jl} = d_0 + t) = \frac{2m - t - 1}{m^2}, \text{ for } t = m, \dots, 2m - 2,$$

where $d_0 = m(d_{il,jl} - 1) + 1$. It is straightforward to verify that

$$E(d'_{il,jl}) = \sum_{t=0}^{2m-2} (d_0 + t) P(d'_{il,jl} = d_0 + t) = m d_{il,jl},$$
(2.5)

$$E((d'_{il,jl})^{2}) = \sum_{t=0}^{2m-2} (d_{0}+t)^{2} P(d'_{il,jl} = d_{0}+t) = m^{2} d^{2}_{il,jl} + \frac{m^{2}-1}{6}.$$
 (2.6)

It is clear that $d_{il,jl} = 0$ when $x_{i,l} = x_{j,l}$. Combining (2.3) and (2.5), we have

$$E(d'_{i,j}) = \sum_{l=1}^{k} E(d'_{il,jl}) = \sum_{l=1}^{k} m d_{il,jl} + (k - h_{i,j})\gamma = m d_{i,j} + (k - h_{i,j})\gamma.$$
(2.7)

Next, combining (2.4) and (2.6), we have

$$E\left(\sum_{l=1}^{k} (d'_{il,jl})^2\right) = \sum_{l=1}^{k} E((d'_{il,jl})^2) = m^2 \sum_{l=1}^{k} d^2_{il,jl} + \frac{m^2 - 1}{6} h_{i,j} + (k - h_{i,j}) \frac{m\gamma}{2}.$$
 (2.8)

Further, we have

$$E((d'_{i,j})^2) = E\left(\left(\sum_{l=1}^k d'_{il,jl}\right)^2\right) = E\left(\sum_{l=1}^k (d'_{il,jl})^2\right) + E\left(\sum_{p\neq q=1}^k d'_{ip,jp}d'_{iq,jq}\right).$$
 (2.9)

Since $d'_{ip,jp}$ and $d'_{iq,jq}$ $(p \neq q)$ are independently determined by the p^{th} and q^{th} columns in the initial design D, with (2.3) and (2.5), we have

$$E\left(\sum_{p\neq q=1}^{k} d'_{ip,jp} d'_{iq,jq}\right) = \sum_{p\neq q=1}^{k} E(d'_{ip,jp}) E(d'_{iq,jq}) = m^{2} \sum_{p\neq q=1}^{k} d_{ip,jp} d_{iq,jq} + 2(k - h_{i,j})\gamma \sum_{l=1}^{k} m d_{il,jl} + (k - h_{i,j})(k - h_{i,j} - 1)\gamma^{2}.$$
 (2.10)

Combining (2.8), (2.9), and (2.10), after some simple algebra, we have

$$Var(d'_{i,j}) = E\left((d'_{i,j})^2\right) - \left(E(d'_{i,j})\right)^2 = C_{1,0} + C_{1,1}h_{i,j}$$

where $C_{1,0}$ and $C_{1,1}$ are constants given in Theorem 1.

Proof of Lemma 2. We need to distinguish two types of operations: level permutation and level expansion. Let σ denote a level permutation and π denote a level expansion. Let E_{σ} denote the expectation toward designs generated by all possible level permutations and E_{π} denote the expectation toward designs generated by all level expansions. As we perform level permutation first and level expansion second, using the properties of conditional expectations, we have

$$E_{\Theta}(d'_{i,j}) = E_{\sigma}[E_{\pi}(d'_{i,j}|\sigma)],$$
 (2.11)

$$Var_{\Theta}(d'_{i,j}) = E_{\sigma}[Var_{\pi}(d'_{i,j}|\sigma)] + Var_{\sigma}[E_{\pi}(d'_{i,j}|\sigma)].$$
 (2.12)

For a given level permutation σ , let $d_{i,j}^{\sigma}$ denote the L_1 -distance of a level permuted design generated by σ . Level permutation does not change pairwise Hamming distances of a design. Applying Theorem 1 to each level permutation σ , we have

$$E_{\pi}(d'_{i,j}|\sigma) = md^{\sigma}_{i,j} + (k - h_{i,j})\gamma, \qquad (2.13)$$

$$Var_{\pi}(d'_{i,j}|\sigma) = C_{1,0} + C_{1,1}h_{i,j}.$$
(2.14)

Similar to the proof of Theorem 1, when considering all possible level permutations,

$$E_{\sigma}(d_{i,j}^{\sigma}) = \frac{s+1}{3}h_{i,j}, \qquad (2.15)$$

$$Var_{\sigma}(d_{i,j}^{\sigma}) = E_{\sigma}((d_{i,j}^{\sigma})^2) - [E_{\sigma}(d_{i,j}^{\sigma})]^2 = \frac{(s+1)(s-2)}{18}h_{i,j}.$$
 (2.16)

Combining (2.11), (2.13), and (2.15), we have

$$E_{\Theta}(d'_{i,j}) = E_{\sigma}[md^{\sigma}_{i,j} + (k - h_{i,j})\gamma] = m\frac{s+1}{3}h_{i,j} + (k - h_{i,j})\gamma$$

= $k\gamma + (m\frac{s+1}{3} - \gamma)h_{i,j}.$ (2.17)

Combining (2.12), (2.13), (2.14), and (2.16), we have

$$Var_{\Theta}(d'_{i,j}) = E_{\sigma}[C_{1,0} + C_{1,1}h_{i,j}] + Var_{\sigma}[md^{\sigma}_{i,j} + (k - h_{i,j})\gamma]$$

$$= C_{1,0} + C_{1,1}h_{i,j} + m^{2}Var_{\sigma}[d^{\sigma}_{i,j}]$$

$$= C_{1,0} + (C_{1,1} + m^{2}\frac{(s+1)(s-2)}{18})h_{i,j}.$$
 (2.18)

Proof of Theorem 2. From (2.17) and (2.18), we have

$$E_{\Theta}\left(\sum_{i\neq j=1}^{n} (d'_{i,j})^{2}\right) = \sum_{i\neq j=1}^{n} E_{\Theta}\left((d'_{i,j})^{2}\right) = \sum_{i\neq j=1}^{n} \left[Var_{\Theta}(d'_{i,j}) + (E_{\Theta}(d'_{i,j}))^{2}\right]$$
$$= \sum_{i\neq j=1}^{n} \left[C_{1,0} + (C_{1,1} + m^{2}\frac{(s+1)(s-2)}{18})h_{i,j}\right] + \sum_{i\neq j=1}^{n} \left[k\gamma + (m\frac{s+1}{3} - \gamma)h_{i,j}\right]^{2}.(2.19)$$

Xu (2003) showed that the GWLP is related to moments of Hamming distances. In particular, for a balanced design with $A_1(D) = 0$, we have the following relationships:

$$\sum_{i \neq j=1}^{n} h_{i,j} = \frac{kn^2(s-1)}{s},$$
(2.20)

$$\sum_{i \neq j=1}^{n} h_{i,j}^2 = \frac{n^2}{s^2} \{ 2A_2(D) + (s-1)k[1 + (s-1)k] \}.$$
 (2.21)

Then the result follows from (2.19), (2.20), and (2.21).

CHAPTER 3

Construction of Maximin Distance Latin Squares and Related Latin Hypercube Designs

3.1 Introduction

Computer experiments are increasingly being used to investigate complex systems (Santner et al., 2013; Fang et al., 2006; Morris and Moore, 2015). The most suitable designs for such experiments are space-filling Latin hypercube designs (Lin and Tang, 2015). Several criteria have been proposed to measure space-filling, including discrepancy criteria via reproducing kernel Hilbert spaces (Hickernell, 1998) and maximin and minimax distance criteria (Johnson et al., 1990). In this paper, we adopt the maximin distance criterion, which maximizes the minimum distance between design points. This criterion optimizes the worst case, thus generating robust space-filling designs. Johnson et al. (1990) showed that maximin distance designs are asymptotically optimal under a Bayesian setting. Morris and Mitchell (1995), Joseph and Hung (2008), Ba et al. (2015) and many others proposed algorithms to construct maximin Latin hypercube designs; see Lin and Tang (2015) for a summary. To the best of our knowledge, the R package SLHD by Ba et al. (2015) is the most efficient current algorithm. Zhou and Xu (2015) proposed to construct maximin Latin hypercube designs via good lattice point sets.

Morris (1991) and Kleijnen (1997) gave many computer models involving several hundred factors, which may require run-economic designs. Under such a situation, it is not unreasonable to assume effect sparsity, that is, relatively few factors are active. Loeppky et al. (2009) provided an informal rule of thumb that the number of runs for a computer experiment should be around ten times the input dimension, but also suggested that, under effect sparsity, the run size should be around ten times the effective dimension, given good a priori knowledge on the number of active factors. In order to identify active factors from a large number of factors with limited budgets or runs, saturated or even supersaturated Latin hypercube designs are useful; see, for example, Butler (2001, 2007). Yet, the construction of such maximin Latin hypercube designs is challenging.

An $n \times n$ Latin square is a supersaturated Latin hypercube design where each row and each column is a permutation of n levels. We propose three algebraic methods for constructing $n \times n$ maximin Latin squares, where n = q, q - 1 or q - 2 and q is a prime or a prime power. We study their properties and derive lower bounds on their minimum distances. The generated Latin squares and related saturated $n \times (n - 1)$ Latin hypercube designs have larger minimum distances than existing ones. Our methods are associated with Costas arrays, which are introduced next.

3.2 Costas Arrays and Welch Method

Costas arrays are widely used in the radar and sonar applications due to their ideal autocorrelation properties (Costas, 1984; Drakakis, 2006). Costas array of order n can be represented geometrically by allocating n points on an $n \times n$ checker-board, such that each row and column has only one point and all of the n(n-1)/2 displacement vectors between each pair of points are distinct. Costas arrays can be represented algebraically as permutation vectors, which are used in this chapter.

Definition 1 (Difference triangle). For any vector $a = (a_1, \ldots, a_n)$, the difference triangle $\mathcal{T}(a)$ is $(t_{i,j})$, where $t_{i,j} = a_{i+j} - a_j$ for $i = 1, \ldots, n-1$ and $j = 1, \ldots, n-i$.

Definition 2 (Costas array). Let $a = (a_1, \ldots, a_n)$ be a permutation of $1, \ldots, n$. Then a is a Costas array of order n if and only if no row in the difference triangle $\mathcal{T}(a)$ contains a repeated value.

Fig. 3.1(a) shows the difference triangle $\mathcal{T}(a)$ for a permutation vector a = (6, 4, 5, 1, 3, 2). All elements in each row of $\mathcal{T}(a)$ are distinct, so a is a Costas array.

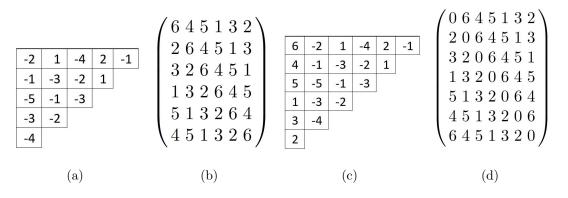


Figure 3.1: Difference triangles and cyclic Latin squares from Costas arrays.
(a) difference triangle for Costas array (6, 4, 5, 1, 3, 2); (b) 6 × 6 cyclic Latin square from Costas array (6, 4, 5, 1, 3, 2); (c) difference triangle for Costas array (0, 6, 4, 5, 1, 3, 2); (d) 7 × 7 cyclic Latin square from Costas array (0, 6, 4, 5, 1, 3, 2).

An $n \times k$ Latin hypercube design is an $n \times k$ matrix where each column is a permutation of n equally-spaced levels, which are denoted by n consecutive numbers, say, 1 to n or 0 to n-1. The minimum distance of a design D, denoted by $d_{\min}(D)$, is the minimum distance between any two distinct rows. In this chapter we consider L_1 -distance, also known as the rectangular or Manhattan distance. For any $n \times k$ Latin hypercube design, the average row pairwise L_1 -distance is (n + 1)k/3 (Zhou and Xu, 2015). The minimum distance cannot exceed the integer part of the average; thus we have the following upper bound.

Lemma 1. For any $n \times k$ Latin hypercube design D, $d_{\min}(D) \leq d_{upper} = \lfloor (n+1)k/3 \rfloor$, where $\lfloor x \rfloor$ is the integer part of x.

Let p be a prime throughout the chapter. In Galois field \mathbb{F}_p , a number α is a primitive root modulo p if and only if for every nonzero element i in \mathbb{F}_p there exists an integer ksuch that $\alpha^k = i \mod p$. In other words, if α is a primitive root modulo p, the vector $(\alpha, \alpha^2, \ldots, \alpha^{p-1}) \mod p$ is a permutation of $1, \ldots, p-1$. The Welch Costas array is defined as follows; see Golomb (1984) and Drakakis (2006).

Definition 3 (Welch Costas array). Let α be a primitive root modulo p. For $i = 1, \ldots, p-1$, let $a_i = \alpha^{i-1+c} \mod p$ where c is an integer and $1 \leq c \leq p-1$. The permutation vector $a = (a_1, \ldots, a_{p-1})$ is a Costas array of order p-1. From a Welch Costas array of order p-1, we can generate a $(p-1) \times (p-1)$ cyclic Latin square by right shifting the vector p-2 times. We can also generate a $p \times p$ cyclic Latin square by augmenting the vector with an additional element 0.

Example 1. For p = 7, the Welch Costas array with primitive root $\alpha = 3$ and parameter c = 3 is a = (6, 4, 5, 1, 3, 2). Fig. 3.1(b) shows the 6×6 cyclic Latin square generated by a. Its minimum distance is 12. To construct a 7×7 Latin square, we use $a_* = (0, a)$ as the generator which is the first row of the cyclic design. Fig. 3.1(c) and (d) show the difference triangle $\mathcal{T}(a_*)$ and the Latin square, respectively. Its minimum distance is 18.

Lemma 2. For any $n \times n$ cyclic Latin square D with generator a, there are at most $\lfloor n/2 \rfloor$ distinct pairwise L_1 -distances, and its ith $(i = 1, ..., \lfloor n/2 \rfloor)$ possible distance is the sum of the absolute values of all elements in the ith and (n-i)th row of the difference triangle $\mathcal{T}(a)$.

As an illustration, the 6×6 cyclic design in Fig. 3.1(b) has 3 possible distances: 14, 12 and 18 which are calculated via the (1st, 5th), (2nd, 4th) and (3rd, 3rd) rows of $\mathcal{T}(a)$ in Fig. 3.1(a), respectively. With Lemma 2, for an $n \times n$ cyclic design, it requires only $O(n^2)$ operations to determine the minimum distance, while for a general $n \times n$ design, it requires $O(n^3)$ operations.

Proposition 1. All possible $(p-1) \times (p-1)$ cyclic Latin squares via generators of Welch Costas arrays with order p-1 are equivalent under row and column permutations.

From the proof of Proposition 1, we can see that all such cyclic Latin squares are equivalent to the leave-one-out good lattice point designs in Zhou and Xu (2015). Thus, the minimum distance of all such designs is $(p^2 - 1)/4$ by Theorem 4 and Proposition 2 in Zhou and Xu (2015).

The $(p-1) \times (p-1)$ Welch designs have bad two-dimensional projections. For example, the points of the Welch design in Fig. 3.1(b) lie on the diagonal when projected onto the first and fourth columns. Here we propose a simple modification: replace p-1 with 0 when constructing $(p-1) \times (p-1)$ Welch designs. The modified Welch designs not only have improved projections and column correlations, but also have larger minimum distances when p > 7, though for p = 5 and p = 7, they have smaller minimum distances. See Section 3.5 for details.

Comparing Fig. 3.1 (a) and (c), $\mathcal{T}(a_*)$ is equivalent to $\mathcal{T}(a)$ adding the Costas array (6, 4, 5, 1, 3, 2) as the first column. Even if a is a Costas array, $a_* = (0, a)$ may or may not be one.

Lemma 3. Let $a_* = (0, a)$. The difference triangle $\mathcal{T}(a_*)$ is equivalent to $\mathcal{T}(a)$ adding the vector a as the first column.

Theorem 1. Let $p \ge 5$ be any prime and a be any Welch Costas array of order p-1. The $p \times p$ cyclic Latin square D with generator $a_* = (0, a)$ has $d_{\min}(D) \ge (p^2 + 7)/8 + 2$.

This bound is very conservative and in practice the results are much better. If α is a primitive root modulo p, $\beta = \alpha^{-1} \mod p$ is another primitive root modulo p. The number of different primitive roots modulo p can be calculated by the Euler's totient function $\phi(p-1)$, which counts the number of integers up to p-1 that are coprime to p-1; $\phi(n) = n \prod_{t|n} (1-1/t)$, where the product is over all distinct prime numbers t dividing n.

Example 2. For p = 7, the primitive roots are 3 and 5. From either primitive root, we can construct six Welch Costas arrays with order 6, and then construct six 7×7 Latin squares. Five designs have $d_{\min} = 16$ and one design has $d_{\min} = 18$. The lower bound in Theorem 1 is 9 and the upper bound in Lemma 1 is 18. The best design from our construction achieves the upper bound, and the worst designs have much larger minimum distance than the lower bound.

Proposition 2. Let α be a primitive root modulo p and $\beta = \alpha^{-1} \mod p$. Let a and b be two Welch Costas arrays with primitive roots α and β , and parameters c_1 and c_2 , respectively. When $c_1 + c_2 = 1 \mod (p - 1)$, the $p \times p$ cyclic Latin squares with generators $a_* = (0, a)$ and $b_* = (0, b)$ have the same distance distribution.

As an illustration, when p = 13, using two Welch Costas arrays with $\alpha = 2$, $c_1 = 8$ and $\beta = 7$, $c_2 = 5$, we can generate two 13×13 designs with the same distance distribution and

minimum distance of 56. Proposition 2 shows that it is equivalent to use primitive root α and $\beta = \alpha^{-1} \mod p$ in the construction. Thus, in all we only need to compare $\phi(p-1)(p-1)/2$ possible designs.

3.3 Gilbert Method

The Gilbert construction was proposed by Gilbert (1965) and called the logarithmic Welch construction by Costas (1984). Gilbert (1965) used these arrays to construct Latin squares without repeated diagrams. Our purpose and use of Gilbert Costas arrays are different from his.

Definition 4 (Gilbert Costas array). Let β be a primitive root modulo p. For i = 1, ..., p-1, let $b_i = \log_{\beta}(i) + 1 - c \mod (p-1)$, where c = 1, ..., p-1; if $b_i = 0$ set $b_i = p-1$. The permutation vector $b = (b_1, ..., b_{p-1})$ is a Costas array of order p-1.

Gilbert Costas arrays are inverse permutations of Welch Costas arrays. As any permutation is a bijection, if $\{f(1), \ldots, f(n)\}$ is a permutation of $\{1, \ldots, n\}$, its inverse permutation is $\{f^{-1}(1), \ldots, f^{-1}(n)\}$.

Example 3. For p = 7, with primitive root 3 and parameter c = 1, the corresponding Welch Costas array is a = (3, 2, 6, 4, 5, 1) and the Gilbert Costas array is b = (6, 2, 1, 4, 5, 3). It is clear that b is the inverse permutation of a. The 6×6 cyclic Latin square with generator b is an equal distance design with all pairwise distances equal to 14. The 7×7 cyclic Latin square with generator $b_* = (0, b)$ has $d_{\min} = 14$.

Theorem 2. Let $p \ge 5$ be a prime and b be a Gilbert Costas array of order p - 1. The $(p-1) \times (p-1)$ cyclic Latin square D with generator b has $d_{\min}(D) \ge (p-1)(p+3)/8$ when $p = 1 \mod 4$, and $d_{\min}(D) \ge (p+1)^2/8$ when $p = 3 \mod 4$.

This lower bound in Theorem 2 is tight for p = 5 or 7. For example, the 6×6 design with generator b = (5, 1, 6, 3, 4, 2), a Gilbert Costas array with primitive root 3 and parameter c = 2, has $d_{\min} = 8$ which equals the lower bound.

Proposition 3. Let a and b be two Gilbert Costas arrays with primitive roots α and β , and parameters c_1 and c_2 , respectively. The $(p-1) \times (p-1)$ cyclic Latin squares with generators a and b have the same distance distribution under either of the following conditions: (i) $\alpha = \beta$ and $c_1 - c_2 = (p-1)/2 \mod (p-1)$; (ii) $\beta = \alpha^{-1} \mod p$ and $c_1 + c_2 = 1 \mod (p-1)$.

By Proposition 3, there are at most $\phi(p-1)(p-1)/4$ designs with different minimum distances via the Gilbert construction. For example, when p = 7, the generated cyclic designs via primitive root 3 and parameters 1, 2, 3, 4, 5 and 6 have the same distance distribution as the designs via primitive root 5 and parameters 6, 5, 4, 3, 2 and 1, and their minimum distances are 14, 8, 12, 14, 8 and 12, respectively.

Theorem 3. Let $p \ge 5$ be a prime and b be a Gilbert Costas array of order p-1. The $p \times p$ cyclic Latin square D with generator $b_* = (0, b)$ has $d_{\min}(D) \ge (p^2 + 7)/4$.

This lower bound in Theorem 3 is roughly 75% of d_{upper} in Lemma 1 for large p, which nearly doubles the lower bound in Theorem 1.

Proposition 4. Let b be any Gilbert Costas array of order p-1 via primitive root β modulo p. All possible $p \times p$ cyclic Latin squares with generators $b_* = (0, b)$ are equivalent under row and column permutations.

By Proposition 4, the $p \times p$ cyclic Latin squares generated via Gilbert Costas arrays do not depend on parameter c. Thus, in all we have $\phi(p-1)$ possible designs.

3.4 Golomb Method

Let $q = p^m$ be a prime power and consider the Galois field \mathbb{F}_q . If m = 1, elements and primitive roots are integers in \mathbb{F}_p . If $m \geq 2$, the elements and primitive roots in \mathbb{F}_q are polynomials. If α is a primitive root, $\alpha^{q-1} = 1$ and $(\alpha, \alpha^2, \ldots, \alpha^{q-1})$ is a permutation vector of nonzero elements in \mathbb{F}_q . There are $\phi(q-1)$ primitive roots in \mathbb{F}_q . Golomb (1984) constructed the following Costas arrays. **Definition 5** (Golomb Costas array). Let α and β be two primitive roots in \mathbb{F}_q where $q = p^m$. For $i, j = 1, \ldots, q - 2$, let $g_i = j$ if $\alpha^i + \beta^j = 1$ in \mathbb{F}_q . The permutation vector $g = (g_1, \ldots, g_{q-2})$ is a Costas array of order q - 2.

The two primitive roots α and β are not necessarily different. By switching α and β , we obtain another Golomb Costas array, which is the inverse permutation.

Theorem 4. Let $q = p^m \ge 7$ be a prime power and g be a Golomb Costas array of order q-2. The $(q-2) \times (q-2)$ cyclic Latin square D with generator g has $d_{\min}(D) \ge q^2/8$ for even q and $d_{\min}(D) \ge (q^2-1)/8$ for odd q.

Theorem 5. Let $q = p^m \ge 7$ be a prime power and g be a Golomb Costas array of order q-2. The $(q-1) \times (q-1)$ cyclic Latin square D with generator $g_* = (0,g)$ has $d_{\min}(D) \ge (q-1)(q-3)/4 + 2$ for odd q and $d_{\min}(D) \ge (q-2)^2/4 + 3$ for even q.

Proposition 5. Given the same primitive root β and possible different α in \mathbb{F}_q , all $(q-1) \times (q-1)$ cyclic Latin squares with generators $g_* = (0,g)$ are equivalent under row and column permutations.

The lower bounds of d_{\min} in Theorems 4 and 5 are roughly 37.5% and 75% of the upper bound d_{upper} in Lemma 1 for large p, respectively. These bounds are conservative and in practice the minimum distances of Golomb designs are much larger.

Example 4. Let $q = 2^4 = 16$ and set the irreducible polynomial as $x^4 + x + 1$ over \mathbb{F}_{16} . Set primitive roots $\alpha = \beta = x$. For i = 1, ..., 14, solving equations $x^i + x^j = 1$ in \mathbb{F}_{16} , we find solution pairs (i, j) which are (1, 4), (2, 8), (3, 14), (6, 13), (11, 12), (7, 9) and (5, 10)where i and j are interchangeable in the solution pairs since $\alpha = \beta$. By Definition 5, this Golomb Costas array is g = (4, 8, 14, 1, 10, 13, 9, 2, 7, 5, 12, 11, 6, 3). The 14×14 Latin square with generator g has minimum distance of 62, and a ratio (d_{\min}/d_{upper}) of 89%. This is much better than the lower bound in Theorem 4 which is 32. The 15×15 Latin square with generator $g_* = (0, g)$ has minimum distance of 70 and a ratio (d_{\min}/d_{upper}) of 88%, where the lower bound in Theorem 5 is 58. Setting m = 1, the Golomb method can efficiently generate $(p - 2) \times (p - 2)$ and $(p - 1) \times (p - 1)$ maximin designs. In generating $(p - 1) \times (p - 1)$ designs, this lower bound in Theorem 5 nearly doubles the lower bound in Theorem 2 where the Gilbert method is used.

Example 5. For p = 13, there are four primitive roots 2, 6, 7 and 11, and thus in all 16 possible Golomb Costas arrays g. With generators g, we can construct four 11×11 designs with $d_{\min} = 38$ and twelve designs with $d_{\min} = 40$. By Proposition 5, with generators $g_* = (0, g)$, we can fix $\alpha = 2$ and there are four possible 12×12 designs whose d_{\min} are 38, 40, 42 and 48, respectively. As a comparison, the best 12×12 Gilbert design has $d_{\min} = 46$.

3.5 Results and Comparisons

In this section, we compare our three methods with the R package SLHD by Ba et al. (2015) and the good lattice point method by Zhou and Xu (2015). The following lemma is straightforward.

Lemma 4. Let D be a Latin square with levels 1 to n and D' be the $(n + 1) \times n$ design obtained by adding a row of zeros to D. Then $d_{\min}(D') = d_{\min}(D)$.

With Lemma 4, we generate $p \times (p-1)$ Latin hypercube designs by adding a row of zeros to our $(p-1) \times (p-1)$ Latin squares from the Welch, Gilbert or Golomb method. Table 3.1 compares $p \times (p-1)$ Latin hypercube designs constructed via different methods and shows their minimum row pairwise L_1 -distances. The $p \times (p-1)$ Welch designs are equivalent to good lattice point designs whereas the modified Welch designs have larger minimum distances than good lattice point designs when p > 7. For the modified Welch designs we add a row of (p-1)'s to the $(p-1) \times (p-1)$ Latin squares whose levels are from 0 to p-2. The Gilbert and Golomb designs outperform good lattice point designs for all cases and outperform linearly permuted good lattice point designs for most cases. For the R package SLHD, we run the command maximinSLHD with option t = 1 and default settings for 100 times, and choose the best results. The best of the Gilbert and Golomb methods are comparable to the R package SLHD, especially for large p. All of our three methods

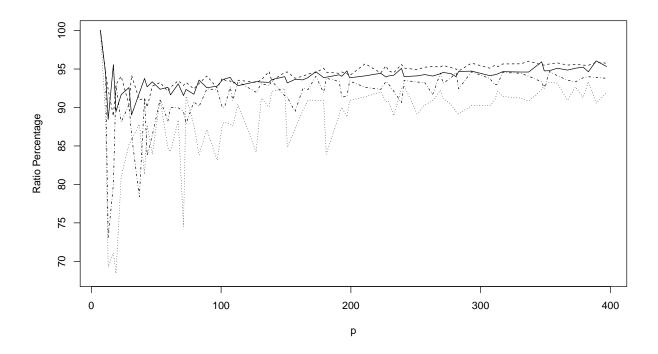


Figure 3.2: Ratio percentages for $(p-1) \times (p-1)$ Latin squares generated by Gilbert method (solid), Golomb method (dashed), simplified Gilbert method (dotted), and simplified Golomb method (dot-dash).

are much faster than the R package SLHD. For example, it takes about an hour for the 97×96 case using the R package SLHD on a laptop with an Intel 2.50GHz I7 CPU, while our algebraic methods take only a few seconds. The minimum distances of our designs can be further improved in some cases by permuting levels as Zhou and Xu (2015) did. We do not pursue this here.

The Gilbert method outperforms the Welch method and the R package SLHD for constructing $p \times p$ Latin hypercube designs when $p \ge 29$, and the Golomb method outperforms the R package SLHD in most cases for constructing $(p-2) \times (p-2)$ Latin hypercube designs; see the Supplementary Material.

Our algebraic construction methods are suitable for constructing high dimensional designs. As p gets larger, the Gilbert and Golomb methods tend to produce better designs in the sense that the ratios of d_{\min}/d_{upper} become higher as shown in Fig. 3.2, where d_{upper} is the upper bound given in Lemma 1. Here we further introduce two simplified methods

р	MWel	Gil	Gol	GLP	LPGLP	SLHD
7	10	14	14	12	13	15
11	32	34	34	30	34	37
13	52	46	48	42	54	50
17	82	86	80	72	84	87
19	104	102	106	90	106	108
23	152	154	158	132	154	159
29	236	250	244	210	250	253
31	268	276	292	240	280	289
37	376	408	404	342	408	411
41	458	512	498	420	508	510
43	502	558	542	462	562	562
47	596	672	668	552	676	672
53	752	848	856	702	846	857
59	926	1056	1050	870	1050	1067
61	988	1134	1130	930	1132	1135
67	1156	1372	1378	1122	1362	1370
71	1328	1518	1538	1260	1516	1541
73	1402	1632	1634	1332	1596	1628
79	1636	1888	1898	1560	1872	1919
83	1802	2122	2112	1722	2090	2120
89	2066	2442	2456	1980	2382	2435
97	2446	2902	2872	2352	2886	2898

Table 3.1: Comparison of L_1 -distance for $p \times (p-1)$ Latin hypercube designs

Note: mWel, modified Welch method; Gil, Gilbert method; Gol, Golomb method; GLP, good lattice point method; LGLP, linearly permuted good lattice point method; SLHD, R package SLHD.

which avoid searching primitive roots and parameters. The simplified Gilbert method uses the smallest primitive root and parameter c = 1. The simplified Golomb method uses the smallest primitive root as α and the second smallest primitive root as β . Fig. 3.2 shows that all of our methods perform well when p is large. The simplified Golomb method is better than the simplified Gilbert method, and the ratios of d_{\min}/d_{upper} are near or above 90% when p > 100 for the former method. It would be interesting to find the explicit forms of d_{\min} for the Gilbert and Golomb methods or to study their asymptotical properties.

From a Latin square, we can generate many Latin hypercube designs by deleting one or more columns. Deleting one column from an $n \times n$ Latin square reduces the minimum distance by at most n - 1. If we start with an $n \times n$ design with large d_{\min}/d_{upper} ratio, we can drop a small number of columns which will lead to good designs with large minimum distances. To drop a comparatively large number of columns, one can adopt a searching scheme such as threshold accepting, which has been thoroughly discussed in Fang et al. (2006).

Based on the Welch, Gilbert and Golomb constructions, there are some secondary constructions of Costas arrays with orders of p, p - 2, p - 3, p^m , $p^m - 1$, $p^m - 3$, $p^m - 4$ and $p^m - 5$; see Beard (2006) and Drakakis et al. (2011). As a generalization of our methods, we can also use these Costas arrays to construct cyclic Latin squares. It is also straightforward to extend all theoretical results in this chapter using the L_2 -distance.

3.6 Appendix

Proof of Lemma 2. Denote $x_1 = (a_1, \ldots, a_n)$ and $x_i = (a_{n-i+2}, \ldots, a_n, a_1, \ldots, a_{n-i+1})$ for $2 \le i \le n$. Let $a_0 = a_n$ for convenience. For any $i < j, x_j$ is obtained from x_i by applying k = j-i steps of right-cyclic shift, and their L_1 -distance is $\sum_{i=1}^n |a_{(i+k) \mod n} - a_i|$ which is denoted as d_k here. Further, $d_{n-k} = \sum_{i=1}^n |a_{(i+n-k) \mod n} - a_i| = \sum_{i=1}^n |a_{(i-k) \mod n} - a_i| = \sum_{i=1}^n |a_{(i+k) \mod n} - a_i| = \sum_{i=1}^n |a_{(i-k) \mod n} - a_i| = \sum_{i=1}^n |a_{(i+k) \mod n} - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} - a_i| + \sum_{i=n-k+1}^n |a_{i+k} \mod n - a_i| = \sum_{i=1}^n |a_{i+k} \mod n - a_i| = \sum_{i=n-k+1}^n |a_{i+k} \mod n$

 $a_{i+k-n} - a_i \mid = \sum_{i=1}^{n-k} \mid a_{i+k} - a_i \mid + \sum_{j=1}^k \mid a_j - a_{n-k+j} \mid = \sum_{i=1}^{n-k} \mid t_{k,i} \mid + \sum_{j=1}^k \mid t_{n-k,j} \mid,$ where $t_{k,j}$ is the *j*th element in the *k*th row of the difference triangle $\mathcal{T}(a)$. This completes the proof.

Proof of Proposition 1. Let $D_{\alpha,c}$ be the $(p-1) \times (p-1)$ generated design using the Welch Costas array with primitive root α and parameter c. Denote the (ith, jth) element in design $D_{\alpha,c}$ where $c \neq 0$ as $x_{i,j}$, and in design $D_{\alpha,0}$ as $y_{i,j}$. Let $j' = j + c \mod (p-1)$ and if j' = 0set j' = p - 1. We have $x_{i,j} = x_{1,j-i+1 \mod (p-1)} = \alpha^{j-i+c \mod (p-1)} \mod p = \alpha^{j'-i \mod (p-1)}$ mod $p = y_{i,j'}$. Thus, any $D_{\alpha,c}$ where $c \neq 0$ is equivalent to $D_{\alpha,0}$ under column permutations. Without loss of generality, let c = 0 in the following proof. For any two different primitive roots α and β , there exists an unique integer t which is coprime to p - 1, such that $\beta = \alpha^{t}$ mod p. Denote the (ith, jth) element in design $D_{\beta,0}$ as $z_{i,j}$. Let $i' = ti \mod (p-1)$ and j' = $tj \mod (p-1)$. Then $j' - i' = t(j-i) \mod (p-1)$ and $\beta^{j'-i'} = \beta^{t(j-i)} = \alpha^{j-i} \mod p$. This leads to $z_{i',j'} = y_{i,j}$. Thus, $D_{\alpha,0}$ and $D_{\beta,0}$ are equivalent under row and column permutations.

Proof of Theorem 1. With Lemmas 2 and 3 and the Costas property of $\mathcal{T}(a)$ that there are no repeated values in any row of the difference triangle, for the $p \times p$ generated design, the lower bound of the pairwise distances can only occur between the 1st and $\{(p+1)/2\}$ th row under the following situations.

(i) When p = 4k + 1 and $k \ge 2$, the lower bound occurs when the (2k)th row of $\mathcal{T}(a)$ consists of numbers: $-1, 1, \ldots, -k, k$, the (2k+1)th row of $\mathcal{T}(a)$ consists of numbers: $-1, 1, \ldots, -(k-1), (k-1), -k$ or k, and the two elements added at the first position are 1 and 2. Under such a situation, by Lemma 2, $d_{\min}(D) \ge 4 \times \{1 + \ldots + (k-1)\} + 3k + 1 + 2 = 2k^2 + k + 3 = (p^2 + 7)/8 + 2$. The bound also holds for p = 5.

(ii) When p = 4k + 3 and $k \ge 1$, the lower bound occurs when the (2k + 1)th row of $\mathcal{T}(a)$ consists of numbers: $-1, 1, \ldots, -(k+1)$ or (k+1), the (2k+2)th row of $\mathcal{T}(a)$ consists of numbers: $-1, 1, \ldots, -k, k$, and the two elements added at the first position are 1 and 2. Under such a situation, by Lemma 2, $d_{\min}(D) \ge 4 \times (1 + \ldots + k) + (k + 1) + 1 + 2 = 2k^2 + 3k + 4 = (p^2 + 7)/8 + 2.$

Proof of Proposition 2. Denote two Welch Costas arrays as a and b where $a_j = \alpha^{j+c_1-1} \mod p$ and $p = \beta^{j+c_2-1} \mod p$. Given $\beta = \alpha^{-1} \mod p$ and $c_1 + c_2 = 1 \mod (p-1)$, $b_j = \alpha^{-j-c_2+1} = \alpha^{p-j-c_2} = \alpha^{p-j+c_1-1} = a_{p-j} \mod p$. Thus, b is the inverse reflection of a. When only considering absolute values and ignoring the order, elements are the same for every uth $(u = 1, \ldots, p-2)$ row of difference triangles $\mathcal{T}(a)$ and $\mathcal{T}(b)$. Define $a_* = (0, a)$ and $b_* = (0, b)$. With Lemma 3, for $\mathcal{T}(a_*)$ and $\mathcal{T}(b_*)$, the sum of the first element of the uth $(u = 1, \ldots, (p-1)/2)$ and (p-u)th rows are the same. Further, by Lemma 2 the $p \times p$ designs with generators of a_* and b_* have the same distance distribution.

Proof of Theorem 3. We first prove a claim that in the difference triangle $\mathcal{T}(b)$, if number v exists in the uth row, $2 \leq u \leq (p-1)/2$, then number -v cannot exist in the (p-u)th row. Suppose otherwise, by Definitions 1 and 4, there exist integers i and j where $1 \leq i, j \leq p-1$, $1 \leq i+u \leq p-1, 1 \leq j+p-u \leq p-1$ and $1 \leq |v| \leq p-1$, such that $\log_{\beta}(i) - \log_{\beta}(i+u) = v \mod (p-1)$ and $\log_{\beta}(j) - \log_{\beta}(j+p-u) = -v \mod (p-1)$. Then, we have $i = (i+u)\beta^{v} \mod p$ and $j + p - u = j\beta^{v} \mod p$. This leads to $ij\beta^{v} = (i+u)(j-u)\beta^{v} \mod p$. Since $\beta^{v} \neq 0 \mod p$, we have $ij = (i+u)(j-u) \mod p$ or $u(j-i-u) = 0 \mod p$. Since $u \neq 0 \mod p$, we have $j = u+i \mod p$. Since $1 \leq j \leq p-1$ and $1 \leq i+u \leq p-1$, we have j = u+i. But for $1 \leq j+p-u \leq p-1$, we have $1 \leq i+p \leq p-1$ which is a contradiction to $1 \leq i \leq p-1$. Thus, our claim is proved.

With Definition 2, Lemma 3 and the proved claim above, ignoring the first column of $\mathcal{T}(b_*)$, for any $u = 2, \ldots, (p-1)/2$, considering the absolute values of elements in the uth and (p-u)th row of $\mathcal{T}(b_*)$ together, no value can appear more than twice. Since β is a primitive root modulo p, $\beta^{(p-1)/2} = p-1 \mod p$ and $\log_{\beta}(p-1) = (p-1)/2$. Then $p-u = (p-1)u \mod p$ and $\log_{\beta}(p-u) = \log_{\beta}(p-1) + \log_{\beta}(u) = (p-1)/2 + \log_{\beta}(u) \mod (p-1)$. This implies $b_{p-u} = b_u + (p-1)/2 \mod (p-1)$. Therefore, with Lemma 2, the lower bound is $d_{\min}(D) \geq 2 \times \{1 + \ldots + (p-3)/2\} + (p-1)/2 + 1 + 1 + (p-1)/2 = (p^2+7)/4$. When considering u = 1, by Definition 2 and Lemma 2, it is straightforward that the above lower bound stands.

Proof of Theorem 5. First we prove a claim that for Golomb Costas array $g = (g_1, \ldots, g_{q-2})$ where $q = p^m$ with primitive roots α and β , in the difference triangle $\mathcal{T}(g)$, if number v exists in the uth row where $2 \leq u \leq (q-1)/2$, then number -v cannot exist in the (q-1-u)th row. Suppose otherwise, by Definition 5 there exist integers i and j where $1 \leq i, j, g_i, g_j \leq q-2$, $1 \leq i+u \leq q-2, 1 \leq j+q-1-u \leq q-2$ and $1 \leq |v| \leq q-2$, such that in Galois field \mathbb{F}_q ,

$$\begin{cases} \alpha^{i} + \beta^{g_{i}} = 1, \\ \alpha^{i+u} + \beta^{g_{i+v}} = 1, \\ \alpha^{j} + \beta^{g_{j}} = 1, \\ \alpha^{j+q-1-u} + \beta^{g_{j-v}} = 1, \end{cases} \Rightarrow \begin{cases} \alpha^{i}\beta^{v} + \beta^{g_{i+v}} = \beta^{v}, \\ \alpha^{i+u} + \beta^{g_{i+v}} = 1, \\ \alpha^{j} + \beta^{g_{j}} = 1, \\ \alpha^{j-u}\beta^{v} + \beta^{g_{j}} = \beta^{v}, \end{cases} \Rightarrow \begin{cases} \alpha^{i}(\beta^{v} - \alpha^{u}) = \beta^{v} - 1, \\ \alpha^{j-u}(\beta^{v} - \alpha^{u}) = \beta^{v} - 1. \\ \alpha^{j-u}\beta^{v} + \beta^{g_{j}} = \beta^{v}, \end{cases}$$

In \mathbb{F}_q , if $\beta^v - \alpha^u \neq 0$, we have $\alpha^i = \alpha^{j-u}$. Since α is primitive root and given the range of i and j, we have j = i+u. Then j+q-1-u = i+q-1 > q-2 contradicts with the condition $1 \leq j+q-1-u \leq q-2$. In \mathbb{F}_q , if $\beta^v - \alpha^u = 0$, we have $\alpha^i + \beta^{c_i} = 1$ and $\alpha^{i+u} + \beta^{c_i+v} = 1$. Thus, we have $\alpha^i + \beta^{c_i} = 1$ and $\alpha^u(\alpha^i + \beta^{c_i}) = 1$. Then, $\alpha^u = 1$ and u = q-1. This contradicts with the range $2 \leq u \leq (q-1)/2$. Thus, our claim is proved. With this claim, similar to the proof of Theorem 3, we can compute the lower bound as follows. When q is odd, $d_{\min}(D) \geq 2 \times \{1 + \ldots + (q-3)/2\} + 1 \times 2 = (q-1)(q-3)/4 + 2$; when q is even, $d_{\min}(D) \geq 2 \times \{1 + \ldots + (q-4)/2\} + (q-2)/2 + 1 + 2 = (q-2)^2/4 + 3$.

3.7 Supplementary Material

3.7.1 Additional Results and Comparisons

In Table 3.2, we show the minimum distances of $p \times p$ Latin hypercube designs generated by our Welch, Gilbert methods and the R package SLHD for $5 \le p < 100$. For the SLHD method, we run the command maximinSLHD with option t = 1 and default settings for 100 times, and choose the best results. It takes about an hour to generate a hundred 97×97 maximin Latin hypercube designs using SLHD on a laptop with an Intel 2.50GHz I7 CPU. In contrast, our algebraic methods take only a few seconds. For p < 29, the Welch method is comparable to the SLHD and is better than the Gilbert method; for $p \ge 29$, the Gilbert method is consistently the best.

	p	Gil	Wel	SLHD	p	Gil	Wel	SLHD
	5	10	10	10	47	708	680	691
	7	16	18	18	53	878	866	877
	11	40	40	41	59	1106	1064	1077
	13	54	56	55	61	1180	1142	1157
	17	92	92	93	67	1426	1368	1395
	19	110	120	115	71	1600	1554	1565
	23	164	166	166	73	1704	1644	1657
	29	266	264	263	79	2000	1892	1941
	31	310	298	300	83	2208	2104	2140
	37	432	420	426	89	2552	2440	2467
	41	534	524	521	97	3030	2866	2938
-	43	584	566	576				

Table 3.2: Comparison of L_1 -distance for $p \times p$ Latin hypercube designs

Note: Gil, Gilbert method; Wel, Welch method; SLHD, R package SLHD.

In Table 3.3, we show the minimum distances of $(p-2) \times (p-2)$ Latin hypercube designs generated by our Golomb method and the SLHD package for $7 \le p < 100$. The Golomb method outperforms the SLHD method for most of the cases, especially for large ones.

3.7.2 Additional Proofs

Proof of Theorem 2. With Lemma 2 and the Costas property of the difference triangle, the lower bound can occur only between the 1st and $\{(p+1)/2\}$ th row of the design. When p = 4k + 1, $d_{min}(D) \ge 4 \times \{1 + \ldots + k\} = (p+3)(p-1)/8$. When p = 4k + 3, $d_{min}(D) \ge 2 \times \{2 \times (1 + \ldots + k) + (k+1)\} = (p+1)^2/8$.

Proof of Proposition 3. (i) Since α is a primitive root, we have $\alpha^{(p-1)/2} = p-1 \mod p$ and $\log_{\alpha}(p-1) = (p-1)/2$. Because $p-i = (p-1)i \mod p$, $\log_{\alpha}(p-i) = \log_{\alpha}(p-1) + \log_{\alpha}(i) = \log_{\alpha}(p-1) + \log_{\alpha}(i)$

~	0.0.		parison	<u> </u>	anoua	100 101 70	, , , , , ,	Herein .	<u>n por our</u>		<u></u>	<u>p</u>
	n	Gol	SLHD	n	Gol	SLHD	n	Gol	SLHD	n	Gol	SLHD
	5	8	10	27	226	229	51	820	805	77	1860	1842
	9	26	28	29	270	263	57	1026	1018	81	2054	2046
	11	40	40	35	386	381	59	1102	1078	87	2388	2351
	15	72	72	39	482	474	65	1314	1311	95	2856	2808
	17	98	92	41	520	525	69	1526	1485			
_	21	140	141	45	638	627	71	1562	1567			

Table 3.3: Comparison of L_1 -distance for $n \times n$ Latin hypercube designs with n = p - 2

Note: Gil, Gilbert method; Gol, Golomb method; GLP, good lattice point method; LP-GLP, linearly permuted good lattice point method.

 $(p-1)/2 + \log_{\alpha}(i) \mod p - 1$. Then $b_i = \log_{\alpha}(i) + 1 - c_2 = \log_{\alpha}(p-i) - (p-1)/2 + 1 - c_2 = \log_{\alpha}(p-i) + 1 - c_1 = a_{p-i} \mod p - 1$. Thus, $b_i = a_{p-i}$, and b is the inverse reflection of a. With Lemma 2, the $(p-1) \times (p-1)$ cyclic Latin squares with generators a and b have the same distance distribution.

(ii) Since $\beta = \alpha^{-1} \mod p$, we have $\log_{\beta}(i) = -\log_{\alpha}(i) \mod p - 1$. Then $b_i = \log_{\beta}(i) + 1 - c_2 = -\log_{\alpha}(i) + 1 - c_2 = -\log_{\alpha}(i) + c_1 = p - a_i \mod p - 1$. Since both a_i and b_i are between 1 and p - 1, we have $b_i = p - a_i$. Then $|b_j - b_i| = |a_j - a_i|$ for any i and j, and the difference triangle of a is the same as that of b towards the absolute values. Therefore, by Lemma 2, the $(p - 1) \times (p - 1)$ cyclic Latin squares with generators a and b have the same distance distribution.

Proof of Proposition 4. Given a primitive root β , denote a and b as the Gilbert Costas arrays with parameters c and c', respectively, where $1 \leq c, c' \leq p-1$. Denote the (*i*th, *j*th) element in the $p \times p$ designs with generators $a_* = (0, a)$ and $b_* = (0, b)$ as $x_{i,j}$ and $y_{i,j}$, respectively. Define $a_0 = b_0 = 0$. We have $x_{i,j} = a_{j-i \mod p}$ and $y_{i,j} = b_{j-i \mod p}$. Let $j' = j\beta^{c'-c} \pmod{p}$ and $i' = i\beta^{c'-c} \pmod{p}$. Then $j' - i' = \beta^{c'-c}(j-i) \mod p$. If i = j, we have $y_{i',j'} = x_{i,j} = 0$. If $i \neq j$, we have $\log_{\beta}(j'-i' \mod p) = \log_{\beta}(j-i \mod p) + (c'-c) \mod p - 1$, which leads to $b_{j'-i' \mod p} = a_{j-i \mod p}$ and $y_{i',j'} = x_{i,j}$. This completes the proof. *Proof of Theorem 4.* By Definition 2 and Lemma 2, similar to the proof of Theorem 2, we obtain the lower bound as follows. When *q* is even, it must be a power of two and $d_{min}(D) \ge 2 \times [2 \times \{1 + 2 + ... + (q - 4)/4\} + q/4] = q^2/8$. When *q* = 1 (mod 4), we have $d_{min}(D) \ge 4 \times \{1 + 2 + ... + (q - 1)/4\} - (q - 1)/4 = (q^2 - 1)/8$; when *q* = 3 (mod 4), we have $d_{min}(D) \ge 4 \times \{1 + 2 + ... + (q - 3)/4\} + (q + 1)/4 = (q^2 - 1)/8$. □

Proof of Proposition 5. Similar to the proof of Proposition 4, let n = q - 1 and $x_{i,j} = g_{j-i \mod n} = \log_{\beta}(1-\alpha^{j-i})$. Let $\alpha = \beta^t$ where t is coprime to n. Then $x_{i,j} = \log_{\beta}(1-\beta^{t(j-i)}) = y_{i',j'}$ where $i' = ti \mod n$ and $j' = tj \mod n$. This shows that the $n \times n$ Latin square generated by α and β is equivalent to the $n \times n$ Latin square generated by β and β . This completes the proof.

CHAPTER 4

Application of Space-Filling Designs in a Drug Combinatorial Experiment on Lung Cancer

4.1 Introduction

Combinatorial drugs have been widely applied in cancer treatments due to its improved efficacy and reduced toxicity compared with individual drugs (Devita et al. (1975) and Ning et al. (2014)). Preclinical experiments in vitro are usually conducted to find the optimal drug combinations. For economic reasons, designs with less runs and good predictive power are preferred. The study of efficient experimental designs and proper follow-up statistical modeling techniques are of much importance. Due to the complexity of underlying biological systems, a systematic quantification of effects for multiple drugs is challenging, and thus various models should be explored for the experiment (Al-Shyoukh et al. (2011)). In such situations, space-filling designs are ideal due to their robustness (Zhou and Xu (2014)).

Several criteria have been proposed to measure designs' space-filling property. Hickernell (1998) proposed the discrepancy criteria via reproducing kernel Hilbert spaces and Johnson et al. (1990) proposed the maximin and minimax distance criteria. In this chapter, we adopt the maximin distance criterion, which maximizes the minimum pair-wise distance between design points. Johnson et al. (1990) showed that maximin distance designs are asymptotically optimal under a Bayesian setting, and are robust since the worst cases are optimized. Moreover, maximin designs are the most suitable for Kriging models (refer to Section 4.3.1).

In this chapter, we focus on a combinatorial drug experiment on lung cancer conducted

Drug	Dosage (μM)							
AG490 (A)	0	0.3	1	3	10	30	100	300
U0126 (B)	0	0.1	0.3	1	3	10	30	100
I-3-M (C)	0	0.3	1	3	10	30	100	300
Coded level	0	1	2	3	4	5	6	7

Table 4.1: Dose levels for each drug in the combinatorial experiment on lung cancer

by Al-Shyoukh et al. (2011). Ning et al. (2014) analyzed the same experiment. Three drugs AG490(A), U0126(B), and indirubin-3'-monoxime (I-3-M)(C) were used and they are inhibitors targeting signaling pathways for cell survival and proliferation. In the experiment, a 512-run 8-level full factorial design ($D_{\rm full}$) was applied to both lung cancer cells and normal cells. The response variable is the ATP level (standardized to 0-1 range) of the cell measured 72 hours after the drug treatment. The actual dosages for each drug are given in Table 4.1 and coded as level 0 to 7.

When analyzing all 512 runs in the experiment, Ning et al. (2014) fitted a Hill-based model with mean square error (MSE) 8.91×10^{-4} for normal cells and 1.42×10^{-2} for cancer cells. Considering the run economy, Al-Shyoukh et al. (2011) recommended 80 random runs to fit linear models and neural networks; under their best model, the MSEs in predicting values for all 512 runs are 6×10^{-3} for normal cells and 1×10^{-2} for cancer cells. In this chapter, we show that when analyzing 512 runs, Kriging model interplates the data which leads to MSE= 0; when using space-filling designs, only 27 runs are needed in Kriging models, and the MSEs in predicting values for all 512 runs are 1.49×10^{-4} for normal cells and 1.29×10^{-3} for cancer cells, which are much smaller than the MSEs in Al-Shyoukh et al. (2011) using 80 runs. In this chapter, we compare four types of designs: a 512-run 8-level full factorial designs, 80-run random sub-designs, 27-run random sub-designs and a 27-run space-filling three-level design under four types of models: Kriging models, neural networks, linear models and Hill-based nonlinear models. We find that it is the best to adopt space-filling designs fitting Kriging models.

This chapter is organized as follows. In Section 4.2, we give the construction of a space-

filling 27-run three-level design. In Section 4.3, we illustrate the Kriging model, neural network, linear model and Hill-based nonlinear model. In Section 4.4, we show the results comparing four types of designs under four types of models. Section 4.5 concludes.

4.2 Construction of space-filling three-level designs

From the 512-run 8-level full factorial design D_{full} , we have 56 choices of three levels for each drug, and thus there are in total 175616 distinct 27-run 3-level sub-designs. After standardizing the actual dosages for each drug to 0-1 range, we search for the space-filling sub-design via a scalar measure (Morris and Mitchell (1995))

$$\phi_p = \left(\sum_{i=2}^n \sum_{j=1}^{i-1} \frac{1}{d_{i,j}^p}\right)^{\frac{1}{p}},\tag{4.1}$$

where p = 15 and $d_{i,j}$ is the L_1 -distance (also known as the rectangular or Manhattan distance) between the i^{th} and j^{th} row of the design. When p is sufficiently large, ϕ_p is asymptotically identical to the maximin distance criterion and smaller ϕ_p value means better space-filling property. The space-filling 3-level sub-design in actual dosages is the one with coded level 0, 6 and 7 for each drug (refer to Table 4.1 for actual dosages). This design is very robust under various models (see Section 4.4); in addition, it enjoys the advantages of being a full factorial design and can give clear interpretations for the main and interaction effects.

4.3 **Response surface modeling**

In this section, we illustrate four types of models to investigate the response surface and give details on how we fit these models for the combinatorial drug experiment.

4.3.1 Kriging models

The Kriging model was proposed by Krige (1951) and systematically introduced in Sacks et al. (1989) and Ginsbourger et al. (2009). The basic idea of Kriging method is that we

can predict the values at target points in the domain via calculating the weighted average of values at known points in the neighborhood. The mathematical form of Kriging model is defined as

$$y(x) = \sum_{j=0}^{L} \beta_j B_j(x) + z(x), \qquad (4.2)$$

where $\{B_j(x), j = 0, ..., L\}$ is a chosen basis over the domain and z(x) is a Gaussian process. In equation (4.2), $\sum_{j=0}^{L} \beta_j B_j(x)$ models the drift of the process mean, which is called the "trend". In this chapter, after comparing various types of trends, we adopt a constant trend model and rely on z(x) to fit the response surface through the observed data. Here z(x) is a Gaussian process with zero mean and covariance

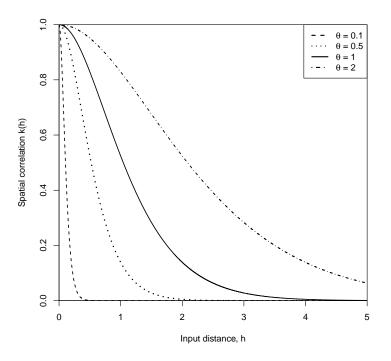
$$C(x_i, x_j) = \sigma^2 * (\prod_{l=1}^d k(x_{i,l}, x_{j,l}) + g\delta_{i,j}),$$

where x_i and x_j are the i^{th} and j^{th} runs, and $x_{i,l}$ and $x_{j,l}$ are the (i^{th}, l^{th}) and (j^{th}, l^{th}) elements in the *d*-dimensional design; σ^2 is a parameter measuring variance; g is the nugget term; $\delta_{i,j} = 0$ if $i \neq j$, otherwise $\delta_{i,j} = 1$. The type of spatial correlation function $k(x_{i,l}, x_{j,l})$ controls the smoothness of the model, and Rasmussen and Williams (2006) and Martin and Simpson (2005) recommended to use the Matérn family. In this chapter, we use the Matérn family

$$k(x_{i,l}, x_{j,l}) = (1 + \frac{\sqrt{5}h}{\theta_l} + \frac{5}{3} * (\frac{h}{\theta_l})^2) * exp(-\frac{\sqrt{5}h}{\theta_l}).$$

where the input distance $h = |x_{i,l} - x_{j,l}|$ and θ_l is the range parameter which scales the correlation length. Figure 4.1 shows some spatial correlation functions with different θ s. For this model, Rasmussen and Williams (2006) showed that data points with higher pairwise L_1 -distance have less correlation. Let y be an unobserved data point. As y moves away from the nearest observations, the variance of this distribution increases. Here we include a small enough nugget effect $g = 10^{-3}$ for 512-run and 80-run designs to avoid numerical problems in model estimation. Under these settings, the parameters are estimated by the maximum likelihood (MLE) method via R package "DiceKriging" (Roustant et al. (2009)). It is ideal to use maximin designs in Kriging models, since any unobserved point will not be too far from the nearest observed point and thus the variance can be controlled.

Figure 4.1: Examples of spatial correlation functions of Matérn family



4.3.2 Neural networks

Al-Shyoukh et al. (2011) fitted a single-layer, four-neuron neural network (perceptron) and a two-layer, single-neuron per layer cascaded neural network. These two models perform similarly and are too simple to perform well. In Section 4.4, we include the former one and an improved neural network in the model comparison. One rule of thumb to choose the number of hidden neurons is

$$N_h = \frac{N_d}{\alpha (N_i + N_o)} \; ,$$

where N_d , N_i and N_o are the number of data, input neurons and output neurons, respectively, and α is a scaling factor from 2 to 10. Since designs used in this chapter have 512, 80 or 27 runs and we want to use the same neural network structure for all these designs in order to make the results comparable, it is proper to adopt a two-layer, four-neuron per layer neural network according to the rule of thumb. Mathematically, for each hidden neuron, the network function

$$f(x) = \frac{1}{1 + e^{-\sum w_i x_i}},$$

where w_i are parameters to be estimated, x_i are input values and *i* ranges from 1 to the number of hidden neurons from the previous layer (including the input layer). For the output neuron $f(x) = \sum w_i x_i$. We fit the neural networks via resilient back-propagation with weight backtracking. We use the R package "neuralnet" (Günther and Fritsch (2010)) with default settings except that we set the number of training repetitions to 10.

4.3.3 Linear and non-linear models

Regression models are the most common analytical tools in drug experiments. In this chapter, we study each type of designs under both linear and non-linear regression models. For the linear model, we include all main, interaction and quadratic effects for drugs A,B and C; the model is

$$y = \beta_0 + \beta_1 A + \beta_2 B + \beta_3 C + \beta_4 A B + \beta_5 A C + \beta_6 B C + \beta_7 A^2 + \beta_8 B^2 + \beta_9 C^2 + \epsilon .$$
(4.3)

For non-linear models, we fit the Hill-based model by Ning et al. (2014). In vivo systems, the relationship between drug dosages and their effects usually follows a sigmoidal curve (Chou (2006)). Based on this, Ning et al. (2014) proposed the Hill-based model

$$y = \frac{1}{1 + (\frac{C}{IC_{50}(\theta)})^{\gamma(\theta)}} + \epsilon , \qquad (4.4)$$

where the total dosage $C = C_1 + C_2 + C_3$, and C_1 , C_2 and C_3 are the actual dosages of drugs A, B and C in the experiment, respectively; the drug proportion $\theta_i = C_i/C$ for i = 1, 2, 3; $IC_{50}(\theta) = a_0 + a_1\theta_1 + a_2\theta_2 + a_3\theta_1\theta_2 + a_4\theta_1^2 + a_5\theta_2^2$; $\gamma(\theta) = b_0 + b_1\theta_1 + b_2\theta_2 + b_3\theta_1\theta_2 + b_4\theta_1^2 + b_5\theta_2^2$. Here function $IC_{50}(\theta)$ measures the dosage of the drug combination which yields 50% effect level, and $\gamma(\theta)$ measures the changing rate of the smooth curve. Hill-based models are able to address all drug combinations and characterize the interaction patterns. When fitting linear models, we standardize the actual dosages of the three drugs to 0-1 range; when fitting Hill-based models, we use the actual dosages without standardization.

4.4 **Results and analysis**

In this section, we compare Kriging models, neural networks, linear models and Hill-based models in fitting four types of designs: the 512-run 8-level full factorial design (D_{full}) , 80-run random sub-design (RD_{80}) , 27-run random sub-design (RD_{27}) and 27-run 3-level space-filling design (MmD_{27}) . For each case, we list the mean square error (MSE) in predicting all 512 runs in the experiment and correlations (r) between predicted values and actual values. We show results for normal cells in Table 4.2 and results for cancer cells in Table 4.3. In all tables, the results for designs RD_{80} and RD_{27} are average values of 100 random ones; results are presented in the format "MSE(r)", and "aE-b" means $a * 10^{-b}$; we round results to 10^{-6} .

			0 0	
	D_{full}	RD_{80}	RD_{27}	MmD_{27}
Kriging	0 (100.00%)	2.09E-04(99.88%)	9.66E-04(99.56%)	1.49E-04(99.95%)
NN1	5.63 E-04 (99.72%)	2.20E-03(98.58%)	3.36E-03(98.27%)	2.86E-03(99.11%)
NN2	7.65 E-05 (99.96%)	1.40E-03(99.30%)	4.63E-03(97.64%)	4.13E-03(98.98%)
Linear	4.97E-04(99.74%)	1.21E-03(99.38%)	4.99E-03(98.00%)	2.94E-03(99.44%)
Hill-based	8.91E-04(99.54%)	8.17E-03(96.53%)*	2.55E-02(88.87%)*	7.64E-03(96.51%)

Table 4.2: Comparison of models and designs in fitting normal cell data

Note: "NN1" represents the single-layer four-neuron neural network; "NN2" represents the twolayer four-neuron per layer neural network; values with "*" are unstable results.

Table 4.3: Comparison of models and designs in fitting cancer cell data

	D_{full}	RD_{80}	RD_{27}	MmD_{27}
Kriging	0 (100%)	3.70E-04(99.78%)	1.84E-03(99.23%)	1.29E-03(99.69%)
NN1	4.72E-04(96.20%)	3.00E-03(98.55%)	4.28E-03(98.27%)	4.14E-03(98.76%)
NN2	6.62E-04(99.72%)	1.75 E-03 (99.26%)	3.63E-03(98.51%)	3.37E-03(98.91%)
Linear	3.02E-03(98.69%)	3.22E-03(98.82%)	4.42E-02(87.69%)	1.70E-02(97.49%)
Hill-based	1.42E-03(99.40%)	1.92E-02(96.74%)*	4.36E-02(86.61%)*	1.33E-03(99.47%)

Note: "NN1" represents the single-layer four-neuron neural network; "NN2" represents the twolayer four-neuron per layer neural network; values with "*" are unstable results. From Tables 4.2 and 4.3, we can see that for both cases of normal cells and cancer cells, Kriging models are the best for all four types of designs in regard to both MSEs and correlations (r). When fitting 512-run design D_{full} , Kriging model interplates the data which leads to MSE=0. When fitting random 80-run data (RD_{80}), the Kriging model gives MSEs $2.09 * 10^{-4}$ for normal cells and $3.07 * 10^{-4}$ for cancer cells, which are also much smaller than the MSEs $6 * 10^{-3}$ for normal cells and $1 * 10^{-2}$ for cancer cells using the best model in Al-Shyoukh et al. (2011). Same conclusions can be drawn if we look at correlations (r).

From Tables 4.2 and 4.3, we can see that for both cases of normal cells and cancer cells, design MmD_{27} outperforms design RD_{27} for all types of models in regard to both MSEs and correlations (r). When fitting the Kriging model which gives the best results, design MmD_{27} performs better than design RD_{80} for normal cells, but worse for cancer cells. Designs MmD_{27} and RD_{80} are comparable in performance. Due to the cost consideration, one would prefer MmD_{27} and this saves 66% of the runs compared with RD_{80} .

In Figures 4.2 and 4.3, we show the scatter-plots of predicted versus observed ATP level (response variable) for both normal and cancer cells using the space-filling three-level design MmD_{27} . From Figure 4.2, it is clear that the Kriging model predicts the rest 485 data points very well and significantly outperforms other models. From Figure 4.3, we can see that both the Kriging and Hill-based model perform well. Note that the purpose of this drug combinatorial experiment is to minimize the ATP level on cancer cells but maximize that on normal cells. From Figure 4.3, it is clear that the Kriging model predicts better for cases with low ATP levels on cancer cells compared with the Hill-based model. Thus, the Kriging model is preferred and design MmD_{27} performs well.

Further, in Figure 4.4, we show the scatter 3D-plot comparing predicted ATP levels between designs D_{full} and MmD_{27} under Kriging models for both normal and cancer cells. In Figure 4.4, values on each dimension are $\log^{(10i)}$ when $i \neq 0$, and 0 when i = 0, where log is the natural logarithm and i is the actual drug dosage. We use colors to represent the ATP level for each drug combination. From Figure 4.4, we can see that the predicted ATP levels via MmD_{27} are nearly the same as the estimated ones via D_{full} . In addition, we also show the contour plots comparing D_{full} and MmD_{27} under Kriging models for both normal

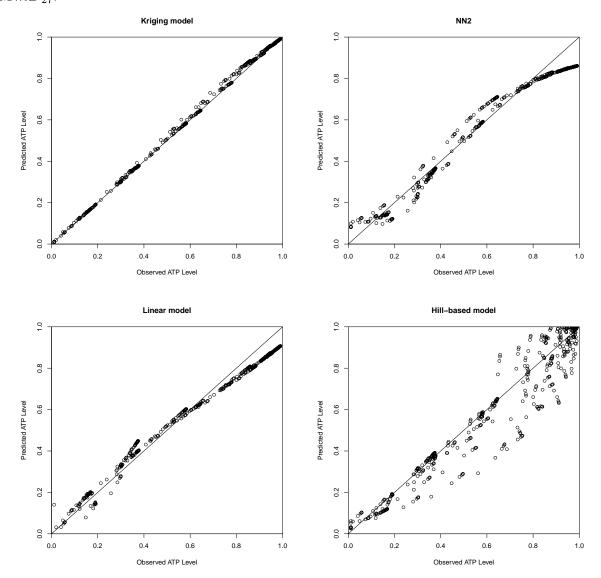


Figure 4.2: Scatter-plots of predicted versus observed ATP levels on normal cells using design MmD_{27} .

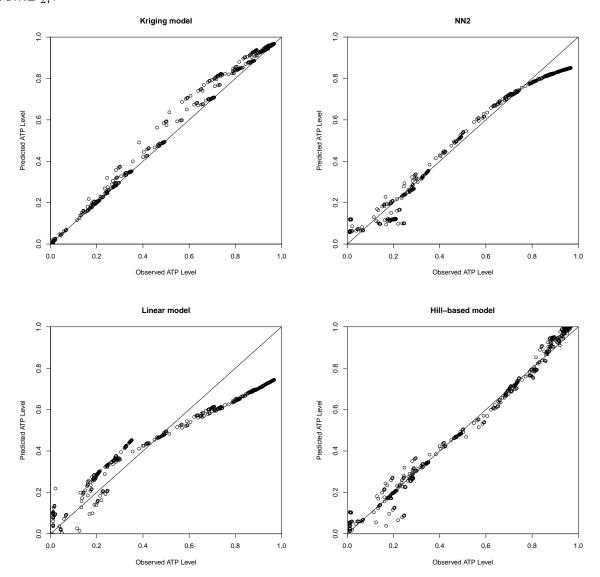


Figure 4.3: Scatter-plots of predicted versus observed ATP levels on cancer cells using design MmD_{27} .

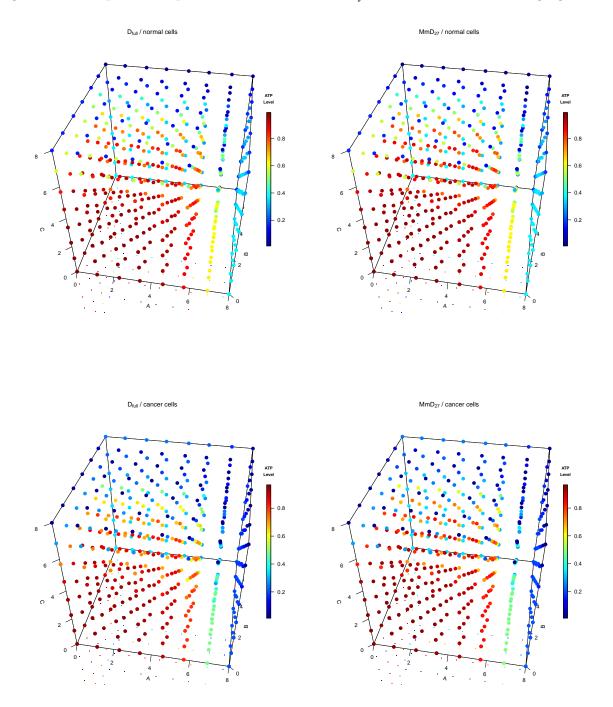


Figure 4.4: Comparison of predicted ATP levels via D_{full} and MmD_{27} under Kriging models.

Figure 4.5: Contour plots of predicted ATP levels via D_{full} and MmD_{27} under Kriging models on normal cells.

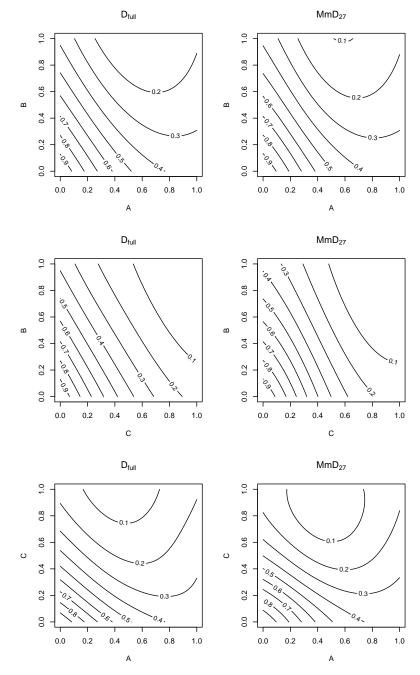
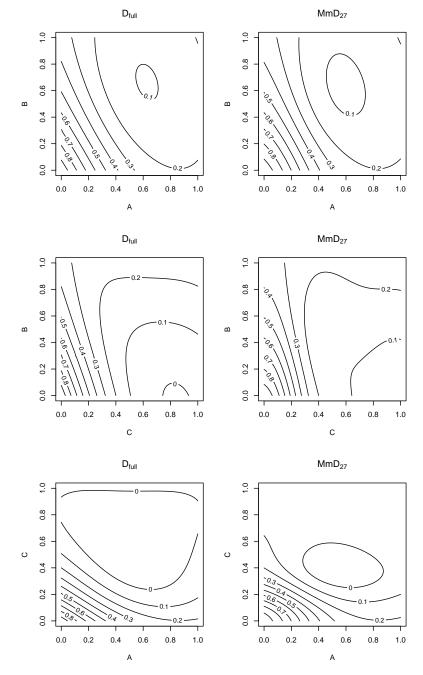


Figure 4.6: Contour plots of predicted ATP levels via D_{full} and MmD_{27} under Kriging models on cancer cells.



and cancer cells. In Figures 4.5 and 4.6, we draw the contour plots for any two drugs while fixing the third to be 0. We can see that for both normal and cancer cells, D_{full} and MmD_{27} performs similarly for all drug combinations of AB, AC and BC. Therefore, design MmD_{27} is good enough to analyze both global response surface and two-drug interactions. In Table 4.4, we give the estimated parameters of the Kriging models using all four types of designs where values for designs RD_{80} and RD_{27} are average values of 100 random ones. In Tables 4.5 and 4.6, we list the five-number summary of estimated parameters for designs RD_{80} and RD_{27} , and we can see that RD_{27} has much larger variations. When using only 27 runs, we need to choose a desirable design, e.g. a space-filling design, rather than a random one.

For normal cells	θ_A	θ_B	θ_C	σ^2	trend
D_{full}	1.23	2.00	1.24	0.26	0.60
MmD_{27}	0.89	1.83	0.77	0.13	0.44
RD_{80}	1.15	1.84	1.13	0.30	0.62
RD_{27}	1.28	2	1.27	0.31	0.69
For cancer cells	θ_A	θ_B	θ_C	σ^2	trend
For cancer cells $D_{\rm full}$	θ_A 1.15		θ_C 0.70		
		1.60		0.26	0.50
D_{full}	1.15	1.60 1.19	0.70	0.26 0.08	0.50

Table 4.4: Estimations of parameters in Kriging models

Furthermore, design MmD_{27} is robust in performance for all four types of models, and its MSEs for both normal and cancer cells are consistently low. On the contrary, design RD_{80} is unstable for Hill-based model (we mark "*" for these values in the table) when trying 100 random designs, and we encountered numeric problems in estimations for two of them. We excluded these results when calculating the average. Design RD_{27} is even more unstable for the Hill-based model. Its average MSEs increase to 10^{-2} level, and we encountered more estimation problems.

	Min	Q1	$\mathbf{Q2}$	Q3	Max
RD_{27}					
θ_A	0.20	1.04	1.13	1.28	1.64
θ_B	0.60	1.78	1.53	2.00	2.00
θ_C	0.67	1.07	1.16	1.23	1.54
σ^2	0.10	0.26	0.30	0.37	0.47
trend	0.46	0.59	0.62	0.67	0.80
RD_{80}					
θ_A	1.10	1.23	1.27	1.33	1.50
θ_B	1.90	2.00	2.00	2.00	2.00
$ heta_C$	1.02	1.22	1.27	1.32	1.42
σ^2	0.23	0.29	0.31	0.34	0.39
trend	0.56	0.64	0.69	0.74	0.84

Table 4.5: Estimations of parameters in Kriging models for normal cells

	Min	Q1	$\mathbf{Q2}$	Q3	Max
RD_{27}					
θ_A	0.20	1.04	1.13	1.28	1.64
θ_B	0.60	1.78	1.53	2.00	2.00
θ_C	0.67	1.07	1.16	1.23	1.54
σ^2	0.10	0.26	0.30	0.37	0.47
trend	0.46	0.59	0.62	0.67	0.80
RD_{80}					
θ_A	1.10	1.23	1.27	1.33	1.50
θ_B	1.90	2.00	2.00	2.00	2.00
$ heta_C$	1.02	1.22	1.27	1.32	1.42
σ^2	0.23	0.29	0.31	0.34	0.39
trend	0.56	0.64	0.69	0.74	0.84

Table 4.6: Estimations of parameters in Kriging models for cancer cells

4.5 Conclusion

In this chapter, we focus on a 512-run combinatorial drug experiment by Al-Shyoukh et al. (2011). We compare four types of designs under four major types of models. We find that when using all 512 runs, the Kriging model outperforms others significantly, including the models used in Al-Shyoukh et al. (2011) and Ning et al. (2014). If the cost of the experiment is considered, it is the best to adopt space-filling designs, e.g., MmD_{27} , and use Kriging models in the follow-up study.

The space-filling three-level design MmD_{27} is not the overall best space-filling design. In total, we have $\binom{512}{27}$ possible 27-run sub-designs, which are impossible to enumerate. There are very limited literature constructing space-filling designs with actual dosages. One possible way is to adopt some global optimization searching algorithm to find space-filling designs, but this requires a lot of time. For a future research, we want to give some algebraic constructions of space-filling designs for combinatorial drug experiments.

CHAPTER 5

Conclusion

Space-filling designs are widely used in computer experiments, combinatorial drug experiments, complex integrals and simulations. There are quite a few literature on how to construct space-filling designs, which can be classified as searching methods and algebraic constructions. The searching methods can generate good designs with flexible sizes, but require much computation and time, especially for large design cases. The algebraic constructions can give good designs using little or no computation, but the sizes of the generated designs are always constrained. In this thesis, we not only propose an efficient searching algorithm, but also give three new algebraic methods to construct space-filling designs. We also show an application of space-filling designs in a combinatorial drug experiments on lung cancer.

In Chapter 2, we propose a computer searching scheme, the MDLE method, to construct maximin designs with flexible run and factor sizes. We justify the method geometrically and prove its efficiency theoretically. We list examples to show that our method outperforms current popular searching algorithms, including the SLHD package (Ba et al., 2015), OMLHD algorithm (Joseph and Hung, 2008), and level permutation method (Zhou and Xu, 2014). In Chapter 3, we propose to construct space-filling designs via Costas arrays. These algebraic methods need very little computation to construct good space-filling Latin square designs and related LHDs. We prove the lower bounds for the minimum L_1 -distances of the generated designs and identify the isomorphic forms. In Chapter 4, we analyze the 512-run 8-level full factorial combinatorial drug experiment by Al-Shyoukh et al. (2011). We show that only 27 runs are needed if we use space-filling designs to fit the Kriging model. The 512 predicted values from the model are nearly the same as the actual values from the experiment, which can be verified via MSEs, correlations, 3-d plots and contour plots.

One key issue of the searching methods in constructing space-filling designs is how to minimize the number of possible designs. As for a future research, we want to study the geometric characteristics of space-filling designs and efficiently shrink the searching space. Our algebraic constructions in Chapter 3 give $q \times q$, $(q-1) \times (q-1)$ and $(q-2) \times (q-2)$ space-filling Latin square designs where q is any prime power. For the future research, we will continue to relax the constraints on the design sizes and give more space-filling designs. Another interesting topic is how to construct space-filling designs with actual dosages or with continuous values. These designs will be very useful in drug combinatorial experiments.

Bibliography

- Al-Shyoukh, I., Yu, F., Feng, J., Yan, K., Dubinett, S., Ho, C.-M., Shamma, J. S., and Sun, R. (2011), "Systematic quantitative characterization of cellular responses induced by multiple signals," *BMC Systems Biology*, 5, 88.
- Ba, S., Myers, W. R., and Brenneman, W. A. (2015), "Optimal sliced Latin hypercube designs," *Technometrics*, 57, 479–487.
- Beard, J. K. (2006), "Generating Costas arrays to order 200," in Information Sciences and Systems, 2006 40th Annual Conference on, IEEE, pp. 1130–1133.
- Box, G. E. and Hunter, J. S. (1961), "The 2^{k-p} fractional factorial designs," *Technometrics*, 3, 311–351.
- Butler, N. A. (2001), "Optimal and orthogonal Latin hypercube designs for computer experiments," *Biometrika*, 847–857.
- (2007), "Supersaturated Latin hypercube designs," Communications in Statistics-Theory and Methods, 34, 417–428.
- Chen, H. (1998), "Some projective properties of fractional factorial designs," Statistics & probability Letters, 40, 185–188.
- Chen, R.-B., Hsieh, D.-N., Hung, Y., and Wang, W. (2013), "Optimizing Latin hypercube designs by particle swarm," *Statistics and Computing*, 23, 663–676.
- Chou, T.-C. (2006), "Theoretical basis, experimental design, and computerized simulation of synergism and antagonism in drug combination studies," *Pharmacological Reviews*, 58, 621–681.
- Cioppa, T. M. and Lucas, T. W. (2007), "Efficient nearly orthogonal and space-filling Latin hypercubes," *Technometrics*, 49, 45–55.
- Costas, J. P. (1984), "A study of a class of detection waveforms having nearly ideal range-Doppler ambiguity properties," *Proceedings of the IEEE*, 72, 996–1009.

- Devita, V. T., Young, R. C., and Canellos, G. P. (1975), "Combination versus single agent chemotherapy: a review of the basis for selection of drug treatment of cancer," *Cancer*, 35, 98–110.
- Drakakis, K. (2006), "A review of Costas arrays," Journal of Applied Mathematics, 2006.
- Drakakis, K., Iorio, F., and Rickard, S. (2011), "The enumeration of Costas arrays of order 28 and its consequences." *Adv. in Math. of Comm.*, 5, 69–86.
- Dueck, G. and Scheuer, T. (1990), "Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing," *Journal of Computational Physics*, 90, 161–175.
- Fang, K.-T., Li, R., and Sudjianto, A. (2006), Design and modeling for computer experiments, CRC Press.
- Fang, K.-T., Shiu, W.-C., and Pan, J.-X. (1999), "Uniform designs based on Latin squares," Statistica Sinica, 905–912.
- Gilbert, E. (1965), "Latin squares which contain no repeated digrams," *Siam Review*, 7, 189–198.
- Ginsbourger, D., Dupuy, D., Badea, A., Carraro, L., and Roustant, O. (2009), "A note on the choice and the estimation of Kriging models for the analysis of deterministic computer experiments," *Applied Stochastic Models in Business and Industry*, 25, 115–131.
- Golomb, S. W. (1984), "Algebraic constructions for Costas arrays," Journal of Combinatorial Theory, Series A, 37, 13–21.
- Günther, F. and Fritsch, S. (2010), "neuralnet: Training of neural networks," The R Journal, 2, 30–38.
- Hickernell, F. (1998), "A generalized discrepancy and quadrature error bound," Mathematics of Computation of the American Mathematical Society, 67, 299–322.

- Johnson, M. E., Moore, L. M., and Ylvisaker, D. (1990), "Minimax and maximin distance designs," Journal of Statistical Planning and Inference, 26, 131–148.
- Joseph, V. R. and Hung, Y. (2008), "Orthogonal-maximin Latin hypercube designs," Statistica Sinica, 171–186.
- Kirkpatrick, S. (1984), "Optimization by simulated annealing: Quantitative studies," Journal of statistical physics, 34, 975–986.
- Kleijnen, J. P. (1997), "Sensitivity analysis and related analyses: a review of some statistical techniques," *Journal of Statistical Computation and Simulation*, 57, 111–142.
- Krige, D. G. (1951), "A statistical approach to some basic mine valuation problems on the Witwatersrand," Journal of the Southern African Institute of Mining and Metallurgy, 52, 119–139.
- Leary, S., Bhaskar, A., and Keane, A. (2003), "Optimal orthogonal-array-based Latin hypercubes," *Journal of Applied Statistics*, 30, 585–598.
- Lin, C. D., Mukerjee, R., and Tang, B. (2009), "Construction of orthogonal and nearly orthogonal Latin hypercubes," *Biometrika*, 243–247.
- Lin, C. D. and Tang, B. (2015), "Latin hypercubes and space-filling designs," in *Handbook* of Design and Analysis of Experiments, Chapman and Hall/CRC, pp. 593–625.
- Loeppky, J. L., Sacks, J., and Welch, W. J. (2009), "Choosing the sample size of a computer experiment: A practical guide," *Technometrics*, 51, 366–376.
- Martin, J. D. and Simpson, T. W. (2005), "Use of Kriging models to approximate deterministic computer models," AIAA Journal, 43, 853–863.
- Moon, H., Dean, A., and Santner, T. (2011), "Algorithms for generating maximin Latin hypercube and orthogonal designs," *Journal of Statistical Theory and Practice*, 5, 81–98.
- Morris, M. D. (1991), "Factorial sampling plans for preliminary computational experiments," *Technometrics*, 33, 161–174.

- Morris, M. D. and Mitchell, T. J. (1995), "Exploratory designs for computational experiments," Journal of Statistical Planning and Inference, 43, 381–402.
- Morris, M. D. and Moore, L. M. (2015), "Design of Computer Experiments: Introduction and Background," in *Handbook of Design and Analysis of Experiments*, Chapman and Hall/CRC, pp. 577–591.
- Ning, S., Xu, H., Al-Shyoukh, I., Feng, J., and Sun, R. (2014), "An application of a Hill-based response surface model for a drug combination experiment on lung cancer," *Statistics in Medicine*, 33, 4227–4236.
- Rasmussen, C. E. and Williams, C. K. (2006), "Gaussian processes for machine learning. 2006," The MIT Press, Cambridge, MA, USA, 38, 715–719.
- Roustant, O., Ginsbourger, D., and Deville, Y. (2009), "The DiceKriging package: Krigingbased metamodeling and optimization for computer experiments," in *Book of abstract of* the R User Conference.
- Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989), "Design and analysis of computer experiments," *Statistical Science*, 409–423.
- Santner, T. J., Williams, B. J., and Notz, W. I. (2013), The design and analysis of computer experiments, Springer Science & Business Media.
- Steinberg, D. M. and Lin, D. K. J. (2006), "A construction method for orthogonal Latin hypercube designs," *Biometrika*, 279–288.
- Sun, F., Liu, M.-Q., and Lin, D. K. J. (2010), "Construction of orthogonal Latin hypercube designs with flexible run sizes," *Journal of Statistical Planning and Inference*, 140, 3236– 3242.
- Tang, B. (1993), "Orthogonal array-based Latin hypercubes," Journal of the American Statistical Association, 88, 1392–1397.

- Tang, Y. and Xu, H. (2013), "An effective construction method for multi-level uniform designs," Journal of Statistical Planning and Inference, 143, 1583–1589.
- Tang, Y., Xu, H., and Lin, D. K. J. (2012), "Uniform fractional factorial designs," The Annals of Statistics, 40, 891–907.
- Winker, P. and Fang, K.-T. (1997), "Application of threshold-accepting to the evaluation of the discrepancy of a set of points," SIAM Journal on Numerical Analysis, 34, 2028–2042.
- Xu, H. (2002), "An algorithm for constructing orthogonal and nearly-orthogonal arrays with mixed levels and small runs," *Technometrics*, 44, 356–368.
- (2003), "Minimum moment aberration for nonregular designs and supersaturated designs," *Statistica Sinica*, 691–708.
- Xu, H., Phoa, F. K. H., and Wong, W. K. (2009), "Recent developments in nonregular fractional factorial designs," *Statistics Surveys*, 3, 18–46.
- Xu, H. and Wu, C. F. J. (2001), "Generalized minimum aberration for asymmetrical fractional factorial designs," *Annals of Statistics*, 549–560.
- (2005), "Construction of optimal multi-level supersaturated designs," The Annals of Statistics, 33, 2811–2836.
- Yang, J. and Liu, M.-Q. (2012), "Construction of orthogonal and nearly orthogonal Latin hypercube designs from orthogonal designs," *Statistica Sinica*, 433–442.
- Zhou, Y. and Xu, H. (2015), "Space-filling properties of good lattice point sets," *Biometrika*, 102, 959–966.
- Zhou, Y.-D. and Xu, H. (2014), "Space-filling fractional factorial designs," Journal of the American Statistical Association, 109, 1134–1144.