Statistica Sinica Preprint No: SS-2024-0379	
Title	Sliced Orthogonal Designs for Computer Experiments
Manuscript ID	SS-2024-0379
URL	http://www.stat.sinica.edu.tw/statistica/
DOI	10.5705/ss.202024.0379
Complete List of Authors	Omar A. Alhelali, S.D. Georgiou and S. Stylianou
Corresponding Authors	S.D. Georgiou
E-mails	stelios.georgiou@rmit.edu.au
Notice: Accepted author version.	

Statistica Sinica

Sliced Orthogonal Designs for Computer Experiments

Omar A. Alhelali[1,2], S.D. Georgiou[1], and S. Stylianou[1]

[1] *RMIT University and* [2] *King Khalid University*

Abstract: Designs for computer experiments are an important class of experimental designs that offer significant cost and time savings over physical experiments. In this paper, we introduce the concept of sliced orthogonal designs for computer experiments, a generalisation of sliced Latin hypercube designs. We propose methods for constructing these sliced orthogonal designs, which form a special class suitable for both first-order and second-order models, with each slice constituting an orthogonal sub-design. The construction methods leverage known sequences with zero autocorrelation function, such as T-sequences and Golay sequences, as well as disjoint amicable sequences. This approach introduces, for the first time, infinite families of such designs. The generated designs are evaluated using various criteria from the literature, and the results are presented in tables for practitioners' reference.

Key words and phrases: Autocorrelation function, Computer experiment, Construction, Integer sequences, Orthogonal designs.

Corresponding Author: Stelios Georgiou, School of Science, RMIT University, Melbourne, VIC 3001, Australia, email: stelios.georgiou@rmit.edu.au

1. Introduction

Computer experiments have become essential tools for modelling and understanding complex real-world systems. Their flexibility has earned widespread recognition in many fields, especially in the physical sciences and engineering Fang et al. (2005); Santner et al. (2003). They offer an efficient and reliable scientific approach to investigate, optimise, compare, and analyse systems, particularly when traditional physical experiments are too costly, time-consuming, or impractical Fang et al. (2005); Santner et al. (2003). Recent works, such as Chen et al. (2018); Wang et al. (2018); Alhelali et al. (2023); Xiao and Xu (2018); Wang et al. (2018); Huang et al. (2021), provide important contributions for those interested in the latest advances in computer experiment design, supporting the ongoing development and refinement of methods in this critical area.

Latin hypercube design (LHD) has attracted considerable interest from the scientific research community and various applied fields, establishing itself as the most widely recognised design for computer experiments. The concept of LHD was first introduced in McKay et al. (1979). A key feature of $LHDs$ is their uniformity across the entire variable space, enabling comprehensive analysis of the variables under study Lin and Tang (2022). An LHD is structured as a matrix with k factors over n runs, where the range of each factor is divided into equal intervals of n, and only one sample is drawn from each interval along each dimension. In other words, each factor has n distinct values drawn from n separate intervals, and each run represents a unique combination of factor values. This design ensures an efficient and diverse exploration of the parameter space.

Quite a lot of work has been done to improve the performance of $LHDs$. For example, maximin $LHDs$, which maximise the minimum distance between design points, have been studied in works such as Morris and Mitchell (1995) and Joseph and Hung (2008). Another significant development is the use of orthogonal array-based $LHDs$, as proposed in Tang (1993). A LHD is considered orthogonal ($OLHD$) when the correlation between any two columns is zero, which means that the factors are not correlated. Important contributions to the study of $OLHDs$ can be found in Georgiou and Efthimiou (2014); Sun and Tang (2017); Li et al. (2021). Furthermore, the use of strong orthogonal arrays to construct $LHDs$ has been explored in studies such as He and Tang (2013); Zhou and Tang (2019); Shi and Tang (2020); Wang et al. (2022), marking another key development in this area.

A $LHD(n, k)$ with $n = qm$ is called a sliced LHD if it can be divided into m slices, each of which is an $LHD(q, k)$ Yang et al. (2016). A Sliced Latin Hypercube Design (SLHD) is a specialised form of Latin Hypercube

Designs, first introduced in Qian (2012). According to their study, SLHDs have two key properties: (1) each individual slice achieves maximal uniformity in any one-dimensional projection, and (2) when all slices are combined, the full design provides the highest level of stratification in one-dimensional projections.

In Qian (2012), the authors proposed $SLHDs$ as effective tools to design computer experiments that involve both quantitative and qualitative variables. In this framework, $SLHD$ is applied to the quantitative factors, with each slice corresponding to a specific combination of levels for the qualitative factors. Given the widespread use of mixed-input computer experiments across scientific and engineering disciplines, sliced designs provide a practical and broadly applicable solution. If a $SLHD$ is orthogonal both as a whole and within each individual slice, it is called an Orthogonal Sliced Latin Hypercube Design ($OSLHD$). The development of $OSLHDs$ has followed several approaches, each with its own advantages and limitations. In Yang et al. (2013), the authors introduced the second-order orthogonal $SLHDs$ with parameters $(2^{2c+1}, 2^c)$, where the number of slices is fixed at 2^r for $r = 1, 2, \ldots, c$. Although these designs are orthogonal, the number of slices cannot be freely chosen, and both the number of runs and the number of factors grow exponentially, creating gaps between available design sizes. For example, when $c = 4$, the design has 512 runs while for $c = 5$, it jumps to 2048 runs.

In Huang et al. (2014), a more general method was proposed, allowing users to create $OSLHDs$, but only if a suitable $OLHDs$ already exists. This makes the method difficult to apply, as it requires finding multiple $OLHDs$ that when projected onto p columns, do not have identical rows - a highly restrictive condition. In addition, the number of runs and factors must match those of the existing designs.

In Cao and Liu (2015), the authors developed $OSLHD(2^{c+1} \cdot t, 2^c, t)$, which allows any number of slices t, but only produces designs where the run size is a power of 2, leaving gaps such as the inability to generate an $OSLHD(24, 12, t)$. In Yang et al. (2016), three different constructions were introduced: the first, $OSLHD(2k \cdot 2^r, 2^r, k)$, and two advanced constructions using the GS-array and the Kharaghani-array, forming $OSLHD(8kp, 4p, k)$ and $OSLHD(16kp, 8p, k)$, respectively. Although these methods offer more flexibility, they require special vectors with zero autocorrelation and specific element properties, which often do not exist.

In Wang et al. (2017), the authors proposed $OSLHD(2^{c+1}, 2^c, s)$, which allows flexibility in the number of slices s but still only produces designs with run sizes that are powers of 2, again making it infeasible to generate intermediate sizes like $OSLHD(24, 12, s)$. In Guo et al. (2023), a more general approach was presented, introducing two algorithms for multilayered designs denoted $SL(s_1, \ldots, s_r; 2m, m)$, where m is the number of factors, $s = s_1 \times \cdots \times s_r$ is the total number of slices and r is the number of layers. However, these designs depend on the existence of orthogonal designs $OD_m(t, a)$ and $OD_m(t, a, b)$, which are only known to exist in a limited number of cases. Although these constructions provide flexibility in the number and structure of slices, they remain constrained by the availability of suitable orthogonal designs.

Finally, in Kumar et al. (2024), a simpler and novel method was proposed for creating $OSLHDs$ with unequal slice sizes, but it only generates designs with four or five slices. For four slices, the method uses $OLHDs$ to generate slices with run sizes $(n_1, n_1 + 1, 2n_1, 4n_1)$; while for five slices, it adds another slice of size $8n_1$. However, this approach is limited to cases where four or five slices are specifically needed and still relies on the existence of suitable $OLHDs$.

A major breakthrough in the field was achieved by Bingham and Sitter Bingham et al. (2009), who fundamentally extended the concept of Latin Hypercube Designs ($LHDs$) by relaxing the traditional constraint that the number of levels must be equal to the number of runs. This innovation marked a turning point in the field, enabling the development of a broader class of designs tailored specifically for computer experiments. Their approach not only generalised $LHDs$ but also laid the foundation for a wide range of subsequent methodologies aimed at constructing more flexible and efficient designs in high-dimensional settings. Further progress in this area was made in Georgiou (2011), where the authors employed Golay sequences and computer-generated vectors to construct new designs for computer experiments. More recent work Alhelali et al. (2023) uses well - known sequences with zero autocorrelation functions—such as T-sequences, Base sequences, Turyn sequences, and others — to construct orthogonal designs for computer experiments with flexible run sizes.

In this paper, we define and introduce a new class of designs called Sliced Orthogonal Designs for Computer Experiments (SOD), which generalises the concept of Sliced Latin Hypercube Designs ($SLHD$). Our approach is motivated by combining the idea of sliced designs — intended to accommodate both quantitative and qualitative variables — with the concept of orthogonal designs for computer experiments, as proposed in Bingham et al. (2009). The designs produced by our method ensure orthogonality of the main effects with terms in both first- and second-order models. Specifically, the overall design and its slices exhibit zero correlation among any odd number of selected columns; for example, the sum of the element-wise product of any three (or any odd number of) columns equals zero.

A key feature of our approach is its flexibility: the number of slices can be any positive integer, offering more options compared to previous constructions such as those in Yang et al. (2013) and Huang et al. (2014). Furthermore, our method supports designs where the number of runs and factors in each slice are multiples of 4. For example, designs such as $SOD(56, 28, m)$ and $SOD(72, 36, m)$ can be generated, whereas, to the best of our knowledge, existing $OSLHD$ methods in the literature cannot produce designs with these parameters.

Moreover, while previous methods (e.g. Yang et al. (2016), Guo et al. (2023)) depend heavily on the existence of orthogonal designs from the literature, our approach constructs the desired designs directly using specific sequences, including T-sequences, Golay sequences, and disjoint amicable sequences Georgiou et al. (2002). As a result, our method is not only straightforward to implement, but also offers broad flexibility in the number of parameters and slices that can be generated.

The remainder of this paper is organised as follows. Section 2 presents preliminary results and key definitions. Section 3 provides evaluations of our designs. Section 4 describes the theorems and steps for constructing

$SODs$. Section 5 presents the designs and additional properties, followed by a discussion and some possible future extensions in Section 6.

2. Definitions and Notation

This section introduces the fundamental concepts and notation necessary to understand the remainder of the paper. A design matrix T is said to be in foldover form if it can be expressed as $T = \binom{D}{-D}$. More details and illustrative examples can be found in Fang et al. (2003).

Let a design D be denoted as $D(n, s^k)$, representing a design for computer experiments with n runs, k factors and s levels per factor, where each level appears with equal frequency. This design is represented by an $n \times k$ matrix $X = [x_1, \ldots, x_k]$, where x_j denotes the column factor j and x_{ij} indicates the level of factor j in the i-th run.

This design extends and generalises the concept of the Sliced Latin Hypercube Design ($SLHD$). In $SLHD$s, the number of levels is equal to the number of runs. In contrast, the proposed design allows the number of levels to be less than or equal to the number of runs ($s \leq n$), with each level replicated equally across the runs. In this work, levels are defined as evenly spaced values. Specifically, for odd s, the levels of factor x_j are given by the set $-(s-1)/2, \ldots, -1, 0, 1, \ldots, (s-1)/2$; for even s, the levels are $-s/2, \ldots, -1, 1, \ldots, s/2$. These levels are uniformly distributed and replicated equally within each column of the design matrix X. Note that when $s = n$, the proposed design coincides with $SLHD$.

Orthogonality is a crucial aspect of experimental design because it allows researchers to efficiently study multiple factors while avoiding confounding effects between them. An orthogonal design for computer experiments is denoted as $OD(n, s^k)$, where n is the number of runs, k is the number of factors, and s is the number of levels for each factor.

Definition 1. If we have an $OD(n, s^k)$, where $n = qm$ and can be divided into m slices (each being an $OD(q, s^k)$), then this design is referred to as a *sliced orthogonal design* for computer experiments, denoted as $SOD(n, s^k, m)$.

In this paper, we introduce a novel design that generalises the concept of the Sliced Orthogonal Latin Hypercube Design ($SOLHD$), previously discussed in Section 1. The key distinction between our proposed design and the original $SOLHD$ lies in the relationship between the number of levels and the number of runs for each factor. Although $SOLHD$s employ an equal number of levels and runs, our design allows for fewer levels than runs. However, the levels in our design are evenly spaced and equally replicated across all factors, preserving desirable statistical properties. This formulation provides a broader framework that includes $SOLHD$s as a special case and enables the construction of designs for parameter settings not supported by existing $SLHD$s.

It is important to note that our approach is not equivalent to merging levels in $SOLHD$, since such merging typically results in the loss of orthogonality and may violate the equal-replication condition. In contrast, our method constructs designs directly, without relying on the constraints inherent to $SOLHD$s. The detailed methodology for constructing these Sliced Orthogonal Designs (SODs) is presented in Section 4.

In regression analysis, it is generally recommended to employ orthogonal independent variables to minimise the risk of confounding effects among predictors, which can distort the estimation of individual regression coefficients. Orthogonality in the design allows the linear effect of each variable to be assessed independently, as it ensures that predictors are not correlated. A design is considered orthogonal when none of its variables exhibit correlation with one another. Incorporating orthogonal designs in regression models improves the precision of coefficient estimation and contributes to more stable and interpretable models.

In response surface methodology, the full second-order model is commonly employed. This polynomial model captures all the effects of linear, quadratic, and two-factor interaction. In such cases, the model can be written as follows:

$$Y = \beta_0 + \sum_{1\le i\le p} \beta_i x_i + \sum_{1\le i\le p} \beta_{ii} x_i^2 + \sum_{1\le i_1<i_2\le p} \beta_{i_1 i_2} x_{i_1} x_{i_2} + \varepsilon,$$

where x_i are the independent variables, β_0 is the intercept, β_i are the coefficients of the linear terms x_i, and β_{ii} represent the coefficients of the quadratic terms x_i^2. The coefficients $\beta_{i_1 i_2}$, for $i_1 \neq i_2$, correspond to the two-factor interaction terms involving $x_{i_1} x_{i_2}$. The term ε denotes the random error.

When constructing orthogonal designs for computer experiments (ODs), it is important to ensure that estimates of linear effects remain uncorrelated. Although second-order effects may also be of interest, we seek designs that satisfy the following two properties: (a) each column of the design is orthogonal to all others and (b) the sum of the element-wise product of any three columns is zero. A design satisfying both conditions is called a *second-order orthogonal design*. It is well known that when a design D employs a foldover structure, it inherently satisfies property (b). In this paper, we ensure that each slice of the generated design satisfies the criteria for second-order orthogonality and define such designs as ODs. This approach guarantees that the overall design maintains second-order orthogonal properties.

Let $A = \{A_j : A_j = (a_{j,0}, a_{j,1}, \ldots, a_{j,n-1}),\ j = 1, \ldots, \ell\}$ be a set of ℓ row vectors, each of length n. The *periodic autocorrelation function* (PAF)

of the set A is defined, with $i+s$ taken modulo n, as

$$P_A(s) = \sum_{j=1}^{\ell}\sum_{i=0}^{n-1} a_{j,i}a_{j,i+s}, \qquad s = 0,1,\ldots,n-1, \tag{2.1}$$

while the *non-periodic autocorrelation function* ($NPAF$) is defined as

$$N_A(s) = \sum_{j=1}^{\ell}\sum_{i=0}^{n-s-1} a_{j,i}a_{j,i+s}, \qquad s = 0,1,\ldots,n-1. \tag{2.2}$$

Example 1. We illustrate the computation of the periodic and non-periodic autocorrelation functions using the following four vectors:

$$A_1 = (1,5,9), \quad A_2 = (-5,1,13), \quad A_3 = (-9,-13,1), \quad A_4 = (-13,9,-5),$$

so that $A = \{A_1, A_2, A_3, A_4\}$.

Periodic Autocorrelation Function (PAF): Using Equation 2.1, we compute:

$$\begin{aligned}
&P_{A_1}(0) = 1^2+5^2+9^2 = 107, && P_{A_1}(1) = 1\cdot 5 + 5\cdot 9 + 9\cdot 1 = 59, && P_{A_1}(2) = 59,\\
&P_{A_2}(0) = 25+1+169 = 195, && P_{A_2}(1) = -5\cdot 1 + 1\cdot 13 + 13\cdot(-5) = -57, && P_{A_2}(2) = -57,\\
&P_{A_3}(0) = 81+169+1 = 251, && P_{A_3}(1) = 117-13-9 = 95, && P_{A_3}(2) = 95,\\
&P_{A_4}(0) = 169+81+25 = 275, && P_{A_4}(1) = -117-45+65 = -97, && P_{A_4}(2) = -97.
\end{aligned}$$

Summing across vectors:

$$P_A(0) = \sum_{j=1}^{4} P_{A_j}(0) = 828, \quad P_A(1) = \sum_{j=1}^{4} P_{A_j}(1) = 0, \quad P_A(2) = \sum_{j=1}^{4} P_{A_j}(2) = 0.$$

Non-Periodic Autocorrelation Function (NPAF): Using Equation 2.2, we compute:

$$
\begin{aligned}
&N_{A_1}(0) = 107, \quad N_{A_1}(1) = 1\cdot 5 + 5\cdot 9 = 50, \qquad N_{A_1}(2) = 1\cdot 9 = 9,\\
&N_{A_2}(0) = 195, \quad N_{A_2}(1) = -5\cdot 1 + 1\cdot 13 = 8, \quad N_{A_2}(2) = -5\cdot 13 = -65,\\
&N_{A_3}(0) = 251, \quad N_{A_3}(1) = 117 - 13 = 104, \qquad N_{A_3}(2) = -9\cdot 1 = -9,\\
&N_{A_4}(0) = 275, \quad N_{A_4}(1) = -117 - 45 = -162, \quad N_{A_4}(2) = -13\cdot(-5) = 65.
\end{aligned}
$$

Summing:

$$N_A(0) = \sum_{j=1}^{4} N_{A_j}(0) = 828,\ N_A(1) = \sum_{j=1}^{4} N_{A_j}(1) = 0,\ N_A(2) = \sum_{j=1}^{4} N_{A_j}(2) = 0.$$

The set of row vectors A is said to have *zero PAF* if $P_A(s) = 0$ for all $s = 1, 2, \ldots, n-1$, and *zero $NPAF$* if $N_A(s) = 0$ for all $s = 1, 2, \ldots, n-1$. In this paper, sequences with zero PAF are sufficient to construct second-order orthogonal designs. However, vectors with zero $NPAF$ offer additional properties and can support a multiplication method that generates new sequences of greater length. Sequences exhibiting zero PAF or $NPAF$ are referred to as *complementary sequences*.

Throughout this paper, we make extensive use of T-sequences and Golay sequences, and we provide a quick overview of basic information on these sequences. T-sequences are sets of four distinct sequences, each having a length of t. These sequences only consist of the values $(-1, 0, 1)$. The key characteristic of T-sequences is that at each position, only one of the four sequences has a nonzero value. Furthermore, the total weight of the T-sequences is t, and they have a zero nonperiodic autocorrelation function ($NPAF$).

It is conjectured that T-sequences exist for all odd lengths (see Conjecture 8.46 in Colbourn and Dinitz (2006)) and many infinite families of such sequences were constructed (see, for example, Colbourn and Dinitz (2006)). The first unresolved case of T-sequences is for length $t = 97$(see Remark 8.47 Colbourn and Dinitz (2006) and Djokovic (2010b,a). All the construction methods in this paper work the same way if we replace the T-sequences needed in the constructions with T-matrices, as these are defined in V2.4 in Colbourn and Dinitz (2006). Let $A = \{A_1, A_2\}$, where $A_1 = (a_{1,1}, a_{1,2}, \ldots, a_{1,n})$ and $A_2 = (a_{2,1}, a_{2,2}, \ldots, a_{2,n})$, be two sequences of length n with each $a_{j,i} \in \{1, -1\}$. If the set satisfies $N_A(s) = N_{A_1}(s) + N_{A_2}(s) = 0$ for all $s = 1, \ldots, n-1$, then the pair A_1 and A_2 are called *Golay sequences* of length n. All Golay sequences of lengths $n = 2^a 10^b 26^c$ exist for any non-negative integers a, b, and c. For more information on T-sequences, Golay sequences, and complementary sequences, we direct the reader to Cohen et al. (1989); Seberry (2017); Seberry and Yamada (1992). In this paper, sequence and vector are used interchangeably, and the choice of term in each context is made to reflect the terminology commonly used in the existing literature.

A circulant matrix of order n is a square matrix created from a vector (sequence) of length n. Each row of the matrix is formed by moving each element of the vector one position to the right compared to the previous row. This shifting action proceeds cyclically, and the final element of the vector used is moved to the first position. Back-circulant matrices have the same shifting pattern but in opposite directions.

Lemma 1. *(Geramita and Seberry,1979, Theorem 4.49). Suppose that there exist four circulant matrices A_1, A_2, A_3, A_4 of order n satisfying. $A_1A_1^T + A_2A_2^T + A_3A_3^T + A_4A_4^T = fI_n$. Then the Goethals-seidel array*

$$GS = \begin{pmatrix} A_1 & A_2R_n & A_3R_n & A_4R_n \\ -A_2R_n & A_1 & A_4^TR_n & -A_3^TR_n \\ -A_3R_n & -A_4^TR_n & A_1 & A_2^TR_n \\ -A_4R_n & A_3^TR_n & -A_2^TR_n & A_1 \end{pmatrix}$$

is a $4n$-order orthogonal matrix, where R_n is the back-diagonal identity matrix of order n.

Corollary 1. *If there are four vectors A, B, C, D of length t with zero periodic autocorrelation function, then these vectors can be used as the first* *rows of circulant matrices, which can be used in Lemma 1 to form an orthogonal matrix of order* $4t$.

Following Kharaghani (2000), a set $\{A_1, A_2, \ldots, A_{2k}\}$ of square real matrices is said to be amicable if $\sum_{i=1}^{k}(A_{2i-1}A_{2i}^T - A_{2i}A_{2i-1}^T) = 0$. We need the following array from Kharaghani (2000).

Lemma 2. *Let* $\{A_i\}_{i=1}^8$ *be an amicable set of circulant matrices of order* n, *satisfying* $\sum_{i=1}^8 A_i A_i^T = fI_n$. *Then, the Kharaghani array*

$$H = \begin{pmatrix} A_1 & A_2 & A_4R_n & A_3R_n & A_6R_n & A_5R_n & A_8R_n & A_7R_n \\ -A_2 & A_1 & A_3R_n & -A_4R_n & A_5R_n & -A_6R_n & A_7R_n & -A_8R_n \\ -A_4R_n & -A_3R_n & A_1 & A_2 & -A_8^TR_n & A_7^TR_n & A_6^TR_n & -A_5^TR_n \\ -A_3R_n & A_4R_n & -A_2 & A_1 & A_7^TR_n & A_8^TR_n & -A_5^TR_n & -A_6^TR_n \\ -A_6R_n & -A_5R_n & A_8^TR_n & -A_7^TR_n & A_1 & A_2 & -A_4^TR_n & A_3^TR_n \\ -A_5R_n & A_6R_n & -A_7^TR_n & -A_8^TR_n & -A_2 & A_1 & A_3^TR_n & A_4^TR_n \\ -A_8R_n & -A_7R_n & -A_6^TR_n & A_5^TR_n & A_4^TR_n & -A_3^TR_n & A_1 & A_2 \\ -A_7R_n & A_8R_n & A_5^TR_n & A_6^TR_n & -A_3^TR_n & -A_4^TR_n & -A_2 & A_1 \end{pmatrix}$$

is an orthogonal matrix of order $8n$.

In the next section, we will quickly review some evaluation criteria that we will use to evaluate our constructed designs, then in Section 4 we will introduce the proposed methods for developing sliced orthogonal designs for computer experiments *SOD*.

3. Evaluation of the generated Designs

This section outlines several criteria for evaluating the quality of experimental designs. A key criterion, introduced by Steinberg and Lin (2006), is the use of *alias matrices* to assess the degree of confounding between second-order effects and a fitted first-order model.

Let X be a design matrix with n runs and k factors, each having s levels. From X, we derive X_1, the model matrix for a first-order model. This matrix is formed by adding a column of ones (representing the intercept) to the k columns of X, resulting in $X_1 = [\mathbf{1}\ X]$.

To assess aliasing with second-order terms, we construct two additional matrices:

- The *interaction matrix* X_{int}, which contains all two-factor interactions. It has dimensions $n \times (k(k-1)/2)$.
- The *quadratic matrix* X_{quad}, which includes all pure quadratic terms. It has dimensions $n \times k$.

The alias matrices quantify the extent to which second-order terms are projected onto the column space of the first-order model. Specifically, the alias matrix for two-factor interactions is given by:

$$T = (X_1^T X_1)^{-1} X_1^T X_{\text{int}},$$

and for the quadratic terms:

$$Q = (X_1^T X_1)^{-1} X_1^T X_{\text{quad}}.$$

These alias matrices are then used to compute two key evaluation metrics: the *average absolute alias* and the *maximum absolute alias*. For the interaction effects, these are defined as follows:

$$\text{ave}(|t|) = E(|t|) = \frac{2\sum_{j=1}^{k+1}\sum_{i=1}^{\frac{k(k-1)}{2}} |t_{ij}|}{m(m^2-1)}, \tag{3.1}$$

$$\max t = \max_{i,j} |t_{ij}|, \tag{3.2}$$

where t_{ij} are the elements of the alias matrix T, and $m = k+1$ is the number of columns in X_1.

Similarly, we evaluate the performance of the generated design in terms of its quadratic terms using the following measures:

$$ave(|q|) = E(|q|) = \frac{2\sum_{j=1}^{k+1}\sum_{i=1}^{k} |q_{ij}|}{m(m+1)} \tag{3.3}$$

$$\max q = \max_{i,j} |q_{ij}| \tag{3.4}$$

If the value of the criterion of interest for design D_1 is less than the value for design D_2, then design D_1 is said to be better than design D_2.

Designs can also be evaluated using a second criterion, *inter-point distances*, to assess how well a design fills the space of interest Morris and

Mitchell (1995). For a given design matrix X, the distance between two points (rows), denoted by s and u, can be measured using the *rectangular distance*, $d_R(s,u)$, or the *Euclidean distance*, $d_E(s,u)$. The rectangular distance is defined as $d_R(s,u) = \sum_{j=1}^{k} |s_j - u_j|$, while the Euclidean distance is given by $d_E(s,u) = \left(\sum_{j=1}^{k} (s_j - u_j)^2\right)^{1/2}$.

Given a design X and a choice of distance metric (rectangular or Euclidean), the set of all interpoint distances is denoted by $(D_1, D_2, \ldots, D_\ell)$, sorted in ascending order, where $\ell = n(n-1)/2$. Let J_i represent the number of pairs of design points that have distance D_i. To construct a *maximin design*, the aim is to maximise the smallest distance D_i while minimising the number of pairs J_i that reach this distance. The criterion is therefore represented by the ordered sequence $(D_1, J_1, D_2, J_2, \ldots, D_i, J_i)$.

For evaluating and ranking designs, we use a scalar summary criterion, where lower values indicate better designs. A commonly used family of functions indexed by a positive integer p is defined as

$$\Phi_p = \left(\sum_{i=1}^{\ell} J_i D_i^{-p} \right)^{1/p}. \tag{3.5}$$

A design that minimises Φ_p is considered a maximin design.

4. The proposed construction methods

We present new strategies for the construction of sliced orthogonal designs for computer experiments using complementary sequences. Unlike traditional methods that directly employ classical orthogonal designs, our approach leverages the structural properties of complementary sequences to generate designs that achieve both orthogonality and sliceability. This construction enables efficient space-filling and supports the integration of both qualitative and quantitative factors in computer experiments.

4.1 Construction Using Goethals–Seidel Arrays

In this section, we introduce two new techniques for generating sliced orthogonal designs for computer experiments (SODs). These constructions are based on the use of Goethals–Seidel orthogonal arrays and circuit matrices. Two well-known classes of complementary sequences, T-sequences and Golay sequences, are used to generate vectors with a zero periodic autocorrelation function (PAF), which serve as building blocks for the desired designs. Once a suitable SOD has been constructed, a level-shifting scheme is applied to adjust the designs entries. This scheme ensures a balanced and symmetric distribution of design points across the levels. The level-shifting rule is defined as follows:

Levels	$1-2i-6m$	$1-2i-4m$	$1-2i-2m$	$1-2i$
Shifted Levels	-7	-5	-3	-1
Levels	$2i-1$	$2i+2m-1$	$2i+4m-1$	$2i+6m-1$
Shifted Levels	1	3	5	7

(4.1)

for $i = 1, 2, \ldots, m$.

4.1.1 Constructions Using T-sequences

This section presents the main idea in Theorem 1, which provides a method for constructing sliced orthogonal designs (SODs) for computer experiments.

Theorem 1. *If there exist T-sequences* (T_1, T_2, T_3, T_4) *of order* t*, then a sliced orthogonal design (SOD) for computer experiments exists with* m *slices,* $8tm$ *runs, and* $4t$ *factors, where* $m = 1, 2, \ldots$

The proof is provided in the Appendix. The following algorithm outlines a step-by-step procedure for constructing SODs using the T-sequences, as shown in Theorem 1. An example is also included to demonstrate the construction process and facilitate understanding.

Algorithm 1 .

Step 1. Generate four vectors with zero periodic (aperiodic) autocorrelation function (PAF) using T-sequences T_1, T_2, T_3, T_4 according to the following expressions:

$$
\begin{aligned}
A_1^i &= (2i-1)T_1 + (2i+2m-1)T_2 + (2i+4m-1)T_3 + (2i+6m-1)T_4 \\
A_2^i &= (1-2i-2m)T_1 + (2i-1)T_2 + (2i+6m-1)T_3 + (1-2i-4m)T_4 \\
A_3^i &= (1-2i-4m)T_1 + (1-2i-6m)T_2 + (2i-1)T_3 + (2i+2m-1)T_4 \\
A_4^i &= (1-2i-6m)T_1 + (2i+4m-1)T_2 + (1-2i-2m)T_3 + (2i-1)T_4
\end{aligned}
\tag{4.2}
$$

Step 2. Use the vectors from Step 1 to construct circulant matrices, and place them into a Goethals–Seidel array as follows:

$$
D_i = \begin{pmatrix}
A_1^i & A_2^i R_t & A_3^i R_t & A_4^i R_t \\
-A_2^i R_t & A_1^i & (A_4^i)^T R_t & -(A_3^i)^T R_t \\
-A_3^i R_t & -(A_4^i)^T R_t & A_1^i & (A_2^i)^T R_t \\
-A_4^i R_t & (A_3^i)^T R_t & -(A_2^i)^T R_t & A_1^i
\end{pmatrix}.
$$

D_i is an orthogonal matrix of size $4t \times 4t$, where $i = 2, \ldots, m$.

Step 3. Apply the foldover technique to generate orthogonal matrices ($T_i = [D_i^T, -D_i^T]^T$). Each T_i represents a single slice in the final SOD.

Step 4. Form the complete design matrix with m slices by stacking all T_i matrices: $X = [T_1^T, T_2^T, \ldots, T_m^T]^T$.

Step 5. Adjust the levels within each slice of X using the level-shifting scheme defined in equation (4.1).

The design constructed by Algorithm 1 is a Sliced Orthogonal Design for Computer Experiments SOD with m slices, $8tm$ runs, and $4t$ factors, where each slice itself also forms an orthogonal design for computer experiments. The following example illustrates how an SOD can be constructed following the steps described in algorithm 1.

Example 2. In this example, we construct a sliced orthogonal design $SOD(48, 12, 2)$ with $m = 2$ slices, where each slice is an orthogonal design $OD(24, 12)$.

Step 1: T-sequences of length $t = 3$ are used. Let

$$T_1 = (1, 0, 0), \quad T_2 = (0, 1, 0), \quad T_3 = (0, 0, 1), \quad T_4 = (0, 0, 0).$$

Substituting these into equation (4.2), we obtain the following vectors with zero PAF for $i = 1, 2$:

For $i = 1$, we have $A_1^1 = (1, 5, 9)$, $A_2^1 = (-5, 1, 13)$, $A_3^1 = (-9, -13, 1)$, $A_4^1 = (-13, 9, -5)$.

For $i = 2$, we have $A_1^2 = (3, 7, 11)$, $A_2^2 = (-7, 3, 15)$, $A_3^2 = (-11, -15, 3)$, $A_4^2 = (-15, 11, -7)$.

Step 2: The vectors from Step 1 are used to construct circulant matrices and placed into the Goethals–Seidel arrays D_1 and D_2 (corresponding to $i = 1, 2$), respectively:

$$D_1 = \begin{pmatrix}
1 & 5 & 9 & 13 & 1 & -5 & 1 & -13 & -9 & -5 & 9 & -13 \\
9 & 1 & 5 & 1 & -5 & 13 & -13 & -9 & 1 & 9 & -13 & -5 \\
5 & 9 & 1 & -5 & 13 & 1 & -9 & 1 & -13 & -13 & -5 & 9 \\
-13 & -1 & 5 & 1 & 5 & 9 & -9 & 5 & 13 & -13 & 1 & -9 \\
-1 & 5 & -13 & 9 & 1 & 5 & 5 & 13 & -9 & 1 & -9 & -13 \\
5 & -13 & -1 & 5 & 9 & 1 & 13 & -9 & 5 & -9 & -13 & 1 \\
-1 & 13 & 9 & 9 & -5 & -13 & 1 & 5 & 9 & -1 & -13 & 5 \\
13 & 9 & -1 & -5 & -13 & 9 & 9 & 1 & 5 & -13 & 5 & -1 \\
9 & -1 & 13 & -13 & 9 & -5 & 5 & 9 & 1 & 5 & -1 & -13 \\
5 & -9 & 13 & 13 & -1 & 9 & 1 & 13 & -5 & 1 & 5 & 9 \\
-9 & 13 & 5 & -1 & 9 & 13 & 13 & -5 & 1 & 9 & 1 & 5 \\
13 & 5 & -9 & 9 & 13 & -1 & -5 & 1 & 13 & 5 & 9 & 1
\end{pmatrix},$$

$$D_2 = \begin{pmatrix}
3 & 7 & 11 & 15 & 3 & -7 & 3 & -15 & -11 & -7 & 11 & -15 \\
11 & 3 & 7 & 3 & -7 & 15 & -15 & -11 & 3 & 11 & -15 & -7 \\
7 & 11 & 3 & -7 & 15 & 3 & -11 & 3 & -15 & -15 & -7 & 11 \\
-15 & -3 & 7 & 3 & 7 & 11 & -11 & 7 & 15 & -15 & 3 & -11 \\
-3 & 7 & -15 & 11 & 3 & 7 & 7 & 15 & -11 & 3 & -11 & -15 \\
7 & -15 & -3 & 7 & 11 & 3 & 15 & -11 & 7 & -11 & -15 & 3 \\
-3 & 15 & 11 & 11 & -7 & -15 & 3 & 7 & 11 & -3 & -15 & 7 \\
15 & 11 & -3 & -7 & -15 & 11 & 11 & 3 & 7 & -15 & 7 & -3 \\
11 & -3 & 15 & -15 & 11 & -7 & 7 & 11 & 3 & 7 & -3 & -15 \\
7 & -11 & 15 & 15 & -3 & 11 & 3 & 15 & -7 & 3 & 7 & 11 \\
-11 & 15 & 7 & -3 & 11 & 15 & 15 & -7 & 3 & 11 & 3 & 7 \\
15 & 7 & -11 & 11 & 15 & -3 & -7 & 3 & 15 & 7 & 11 & 3
\end{pmatrix}.$$

Step 3: Apply the foldover technique to the matrices D_1 and D_2 as follows: $T_1 = [D_1^T, -D_1^T]^T$ and $T_2 = [D_2^T, -D_2^T]^T$. Each matrix T_i is an orthogonal design for computer experiments.

Step 4: Construct the final sliced orthogonal design $SOD(48, 12, 2)$ with two slices by stacking the slices: $X = [T_1^T, T_2^T]^T$.

Step 5: Finally, the level values in each slice are adjusted using the level-shifting scheme defined earlier in equation (4.1). For this example, the mapping is shown in the table below:

Levels	$-15, -13$	$-11, -9$	$-7, -5$	$-3, -1$	$1, 3$	$5, 7$	$9, 11$	$13, 15$
Shifted Levels	-7	-5	-3	-1	1	3	5	7

4.1.2 Construction Using Golay Sequences

This section describes how Golay sequences can be used to construct sliced orthogonal designs (SODs) for computer experiments. These sequences help generate designs with the required orthogonality and balance properties.

Theorem 2. *If there exist Golay sequences of order ℓ, then a sliced orthogonal design (SOD) for computer experiments exists with m slices, $8nm$ runs, and $4n$ factors, where $m = 1, 2, \ldots$ and $n = 2\ell + 1$.*

The proof is provided in the Appendix. The following algorithm outlines a step-by-step procedure for constructing an SOD using Golay sequences, as stated in Theorem 2. This method supports the generation of designs for different values of m and for various Golay sequences.

Algorithm 2 Construction of a Sliced Orthogonal Design using Golay Sequences

Let A_1 and A_2 be Golay sequences of length ℓ.

Step 1: Construct the vectors X_1, X_2, and X_3 of length $n = 2\ell + 1$ as follows:

$$X_1 = \{1, \underbrace{0, \ldots, 0}_{2\ell}\}, \quad X_2 = \{0, A_1, \underbrace{0, \ldots, 0}_{\ell}\}, \quad X_3 = \{\underbrace{0, \ldots, 0}_{\ell+1}, A_2\}$$

Then, for $i = 1, 2, \ldots, m$, compute:

$$\begin{aligned}
A_1^i &= (2i-1)X_1 + (2i+2m-1)X_2 + (2i+4m-1)X_3 \\
A_2^i &= (1-2i-2m)X_1 + (2i-1)X_2 + (2i+6m-1)X_3 \\
A_3^i &= (1-2i-4m)X_1 + (1-2i-6m)X_2 + (2i-1)X_3 \\
A_4^i &= (1-2i-6m)X_1 + (2i+4m-1)X_2 + (1-2i-2m)X_3
\end{aligned} \tag{4.3}$$

Step 2: Use the vectors $A_1^i, A_2^i, A_3^i, A_4^i$ to construct the Goethals–Seidel matrix D_i:

$$D_i = \begin{pmatrix} A_1^i & A_2^i R_n & A_3^i R_n & A_4^i R_n \\ -A_2^i R_n & A_1^i & (A_4^i)^T R_n & -(A_3^i)^T R_n \\ -A_3^i R_n & -(A_4^i)^T R_n & A_1^i & (A_2^i)^T R_n \\ -A_4^i R_n & (A_3^i)^T R_n & -(A_2^i)^T R_n & A_1^i \end{pmatrix}$$

Step 3: Apply the foldover technique: $\quad T_i = [D_i^T, -D_i^T]^T$

Step 4: Form the complete design with m slices: $X = [T_1^T, T_2^T, \ldots, T_m^T]^T$.

Step 5: Adjust the levels in each slice of X using the shifting scheme defined in equation (4.1).

Using Golay sequences of length ℓ, Algorithm 2 constructs a sliced orthogonal design for computer experiments (SOD) with m slices, $8nm$ runs and $4n$ factors, where $n = 2\ell + 1$. Each slice forms an orthogonal design. To illustrate the construction process, the following example demonstrates how a specific SOD can be generated using Algorithm 2.

Example 3. In this example, we construct a sliced orthogonal design $SOD(80, 20, 2)$ using Algorithm 2, with $m = 2$ slices, each corresponding to an orthogonal design $OD(40, 20)$.

Step 1: Let $A_1 = (1, 1)$ and $A_2 = (1, -1)$ be Golay sequences of length $\ell = 2$. Then $n = 2\ell + 1 = 5$. The base vectors are defined as:

$$X_1 = (1, 0, 0, 0, 0), \quad X_2 = (0, 1, 1, 0, 0), \quad X_3 = (0, 0, 0, 1, -1).$$

Using equation (4.3), we generate the following vectors with zero PAF for $i = 1, 2$:

$$\begin{aligned} \text{For } i = 1: \quad & A_1^1 = (1, 5, 5, 9, -9), & A_2^1 = (-5, 1, 1, 13, -13), \\ & A_3^1 = (-9, -13, -13, 1, -1), & A_4^1 = (-13, 9, 9, -5, 5). \end{aligned}$$

$$\begin{aligned} \text{For } i = 2: \quad & A_1^2 = (3, 7, 7, 11, -11), & A_2^2 = (-7, 3, 3, 15, -15), \\ & A_3^2 = (-11, -15, -15, 3, -3), & A_4^2 = (-15, 11, 11, -7, 7). \end{aligned}$$

Step 2: The above vectors are used to construct the Goethals–Seidel matrices D_1 and D_2 as described in Step 2 of Algorithm 2. The complete matrices are provided in the Appendix 6.

Step 3: Apply the foldover technique:

$$T_1 = [D_1^T, -D_1^T]^T, \quad T_2 = [D_2^T, -D_2^T]^T.$$

Step 4: The final sliced orthogonal design is:

$$X = [T_1^T, T_2^T]^T.$$

Step 5: Lastly, adjust the levels in each slice using the shifting scheme defined in equation (4.1). The mapping used is shown below:

Levels	$-15, -13$	$-11, -9$	$-7, -5$	$-3, -1$	$1, 3$	$5, 7$	$9, 11$	$13, 15$
Shifted Levels	-7	-5	-3	-1	1	3	5	7

Table 1 presents a range of designs generated using Theorem 1. The first column indicates the number of slices m that can be produced, where m can be any positive integer, offering flexibility in the design construction. It is important to note that for each value of m, different combinations of parameters may arise depending on the chosen length of the T-sequences (t). For example, in the first row of Table 1, $m = 1$ corresponds to a design with a single slice. When the T-sequence length is $t = 1$, the resulting design is $SOD(8, 4)$, which also satisfies the conditions of a sliced Latin hypercube design (SLHD). Increasing t to 3 yields a larger design, $SOD(24, 12)$, etc. This shows the flexibility of the method, allowing for variation even within a fixed number of slices.

Table 1: Designs constructed by Theorem 1

Number of	**Length of T-Sequences (t)**				
Slices (m)	1	3	5	$\cdots$	t
1	$SOD(8, 4)$	$SOD(24, 12)$	$SOD(40, 20)$	$\cdots$	$SOD(8t, 4t)$
2	$SOD(16, 4)$	$SOD(48, 12)$	$SOD(80, 20)$	$\cdots$	$SOD(16t, 4t)$
3	$SOD(24, 4)$	$SOD(72, 12)$	$SOD(120, 20)$	$\cdots$	$SOD(24t, 4t)$
4	$SOD(32, 4)$	$SOD(96, 12)$	$SOD(160, 20)$	$\cdots$	$SOD(32t, 4t)$
5	$SOD(40, 4)$	$SOD(120, 12)$	$SOD(200, 20)$	$\cdots$	$SOD(40t, 4t)$
6	$SOD(48, 4)$	$SOD(144, 12)$	$SOD(240, 20)$	$\cdots$	$SOD(48t, 4t)$
7	$SOD(56, 4)$	$SOD(168, 12)$	$SOD(280, 20)$	$\cdots$	$SOD(56t, 4t)$
8	$SOD(64, 4)$	$SOD(192, 12)$	$SOD(320, 20)$	$\cdots$	$SOD(64t, 4t)$
9	$SOD(72, 4)$	$SOD(216, 12)$	$SOD(360, 20)$	$\cdots$	$SOD(72t, 4t)$
10	$SOD(80, 4)$	$SOD(240, 12)$	$SOD(400, 20)$	$\cdots$	$SOD(80t, 4t)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
m	$SOD(8m, 4)$	$SOD(24m, 12)$	$SOD(40m, 20)$	$\cdots$	$SOD(80tm, 4t)$

4.2 Construction Using the Kharaghani Array

In this section, we introduce an alternative approach for constructing sliced orthogonal designs for computer experiments, based on the Kharaghani array. This method incorporates Golay sequences and disjoint amicable sequences that were defined in Georgiou et al. (2002). After generating the desired design, a new level-shifting scheme will be applied as described below.

levels	$1-16i$	$3-16i$	$5-16i$	$7-16i$	$9-16i$	$11-16i$	$13-16i$	$15-16i$
Shifted levels	-15	-13	-11	-9	-7	-5	-3	-1
levels	$16i-15$	$16i-13$	$16i-11$	$16i-9$	$16i-7$	$16i-5$	$16i-3$	$16i-1$
Shifted levels	1	3	5	7	9	11	13	15

(4.4)

for $i = 1, 2, ..., m$.

In the following theorem, we employ amicable disjoint sequences, as listed in Table 1 of Georgiou et al. (2002), to construct a new class of sliced orthogonal designs for computer experiments.

Theorem 3. *Let* X_1 *and* X_2 *be* $\{0, \pm 1\}$ *circulant matrices of order* n *satisfying:*

$$X_1 X_1^T + X_2 X_2^T = f I_n, \quad X_1 X_2^T - X_2 X_1^T = 0, \quad X_1 * X_2 = 0,$$

where $*$ denotes the Hadamard (elementwise) product of the matrices. Then there exists a Sliced Orthogonal Design for Computer Experiments (SOD) with m slices, $16nm$ runs, and $8nl$ factors, for any $m = 1, 2, \ldots$.

The proof is provided in the Appendix. The following algorithm outlines the construction of a SOD using amicable disjoint sequences and the Kharaghani array.

Algorithm 3 Construction of SOD using the Kharaghani array and disjoint amicable sequences

Step 1: Let X_1 and X_2 be disjoint amicable sequences of length n. For each $i = 1, \ldots, m$, construct the following eight vectors:

$$\begin{aligned}
A_1^i &= (16i-15)X_1 + (16i-13)X_2 & A_5^i &= (16i-7)X_1 + (16i-5)X_2 \\
A_2^i &= (13-16i)X_1 + (16i-15)X_2 & A_6^i &= (5-16i)X_1 + (16i-7)X_2 \\
A_3^i &= (16i-11)X_1 + (16i-9)X_2 & A_7^i &= (16i-3)X_1 + (16i-1)X_2 \\
A_4^i &= (9-16i)X_1 + (16i-11)X_2 & A_8^i &= (1-16i)X_1 + (16i-3)X_2
\end{aligned} \tag{4.5}$$

Algorithm 3 (cont.)

Step 2: Construct the Kharaghani matrix D_i using the vectors above as circulants: $D_i =$

$$\begin{pmatrix} A_1^i & A_2^i & A_4^iR_n & A_3^iR_n & A_6^iR_n & A_5^iR_n & A_8^iR_n & A_7^iR_n \\ -A_2^i & A_1^i & A_3^iR_n & -A_4^iR_n & A_5^iR_n & -A_6^iR_n & A_7^iR_n & -A_8^iR_n \\ -A_4^iR_n & -A_3^iR_n & A_1^i & A_2^i & -(A_8^i)^TR_n & (A_7^i)^TR_n & (A_6^i)^TR_n & -(A_5^i)^TR_n \\ -A_3^iR_n & A_4^iR_n & -A_2^i & A_1^i & (A_7^i)^TR_n & (A_8^i)^TR_n & -(A_5^i)^TR_n & -(A_6^i)^TR_n \\ -A_6^iR_n & -A_5^iR_n & (A_8^i)^TR_n & -(A_7^i)^TR_n & A_1^i & A_2^i & -(A_4^i)^TR_n & (A_3^i)^TR_n \\ -A_5^iR_n & A_6^iR_n & -(A_7^i)^TR_n & -(A_8^i)^TR_n & -A_2^i & A_1^i & (A_3^i)^TR_n & (A_4^i)^TR_n \\ -A_8^iR_n & -A_7^iR_n & -(A_6^i)^TR_n & (A_5^i)^TR_n & (A_4^i)^TR_n & -(A_3^i)^TR_n & A_1^i & A_2^i \\ -A_7^iR_n & A_8^iR_n & (A_5^i)^TR_n & (A_6^i)^TR_n & -(A_3^i)^TR_n & -(A_4^i)^TR_n & -A_2^i & A_1^i \end{pmatrix}$$

Step 3: Apply the foldover technique to generate orthogonal slices:

$$T_1 = [D_1^T, -D_1^T]^T, \quad T_2 = [D_2^T, -D_2^T]^T, \quad \ldots, \quad T_m = [D_m^T, -D_m^T]^T$$

Step 4: Form the final sliced orthogonal design: $X = [T_1^T, T_2^T, \ldots, T_m^T]^T$.

Step 5: Adjust the levels in each slice using the shifting scheme defined in equation (4.4).

The algorithm 3 constructs a sliced orthogonal design SOD with m slices, $16nm$ runs, and $8n$ factors. Each slice independently forms an orthogonal design for computer experiments.

Remark 1. Several examples of disjoint amicable sequences $\{X_1, X_2\}$ are provided in Table 1 of Georgiou et al. (2002). The level distribution in the resulting design depends on the relationship between the length and weight (# of non-zero elements) of the sequences, leading to two cases:

- If Length = Weight, the levels are replicated uniformly.

- If Length $\neq$ Weight, all levels are still replicated uniformly, except that level zero appears more frequently. In this case, the following vectors replace those in Step 1 of Algorithm 3:

$$A_1^i = (8i-7)X_1 + (8i-6)X_2 \qquad A_5^i = (8i-3)X_1 + (8i-2)X_2$$
$$A_2^i = (6-8i)X_1 + (8i-7)X_2 \qquad A_6^i = (2-8i)X_1 + (8i-3)X_2$$
$$A_3^i = (8i-5)X_1 + (8i-4)X_2 \qquad A_7^i = (8i-1)X_1 + (8i)X_2$$
$$A_4^i = (4-8i)X_1 + (8i-5)X_2 \qquad A_8^i = (-8i)X_1 - (8i-1)X_2$$

After the design is constructed, the following scheme ensures even level distribution:

Levels	$-8i$	$1-8i$	$2-8i$	$3-8i$	$4-8i$	$5-8i$	$6-8i$	$7-8i$
Shifted Levels	-8	-7	-6	-5	-4	-3	-2	-1
Levels	$8i-7$	$8i-6$	$8i-5$	$8i-4$	$8i-3$	$8i-2$	$8i-1$	$8i$
Shifted Levels	1	2	3	4	5	6	7	8

for $i = 1, 2, \ldots, m$.

Example 4. This example demonstrates how to construct $SOD(32, 8, 2)$ using Theorem 3, with $m = 2$ slices where each slice is an $OD(16, 8)$.

Step 1: Select disjoint amicable sequences from Table 1 of Georgiou et al. (2002) with $n = 1$: $X_1 = (1)$ and $X_2 = (0)$. Using these in equation (4.5), we generate:

$$A_1^1 = 1,\ A_2^1 = -2,\ A_3^1 = 3,\ A_4^1 = -4,\ A_5^1 = 5,\ A_6^1 = -6,\ A_7^1 = 7,\ A_8^1 = -8$$

$$A_1^2 = 9,\ A_2^2 = -10,\ A_3^2 = 11,\ A_4^2 = -12,\ A_5^2 = 13,\ A_6^2 = -14,\ A_7^2 = 15,\ A_8^2 = -16$$

Step 2: Construct the circulant matrices D_1 and D_2 using these vectors. Since $n = 1$, where the D_1 and D_2 are the 8×8 matrices:

$$\begin{pmatrix} 1 & -3 & -7 & 5 & -11 & 9 & -15 & 13 \\ 3 & 1 & 5 & 7 & 9 & 11 & 13 & 15 \\ 7 & -5 & 1 & -3 & 15 & 13 & -11 & -9 \\ -5 & -7 & 3 & 1 & 13 & -15 & -9 & 11 \\ 11 & -9 & -15 & -13 & 1 & -3 & 7 & 5 \\ -9 & -11 & -13 & 15 & 3 & 1 & 5 & -7 \\ 15 & -13 & 11 & 9 & -7 & -5 & 1 & -3 \\ -13 & -15 & 9 & -11 & -5 & 7 & 3 & 1 \end{pmatrix}, \begin{pmatrix} 17 & -19 & -23 & 21 & -27 & 25 & -31 & 29 \\ 19 & 17 & 21 & 23 & 25 & 27 & 29 & 31 \\ 23 & -21 & 17 & -19 & 31 & 29 & -27 & -25 \\ -21 & -23 & 19 & 17 & 29 & -31 & -25 & 27 \\ 27 & -25 & -31 & -29 & 17 & -19 & 23 & 21 \\ -25 & -27 & -29 & 31 & 19 & 17 & 21 & -23 \\ 31 & -29 & 27 & 25 & -23 & -21 & 17 & -19 \\ -29 & -31 & 25 & -27 & -21 & 23 & 19 & 17 \end{pmatrix}$$

Step 3: Apply the foldover technique:

$$T_1 = \begin{bmatrix} D_1^T & -D_1^T \end{bmatrix}^T, \quad T_2 = \begin{bmatrix} D_2^T & -D_2^T \end{bmatrix}^T$$

Step 4: The final $SOD(32, 8, 2)$ with 2 slices is:

$$X = \begin{bmatrix} T_1^T & T_2^T \end{bmatrix}^T$$

Step 5: The shifting scheme for this example is:

Levels	$-31, -15$	$-29, -13$	$-27, -11$	$-25, -9$
Shift Levels	-15	-13	-11	-9
Levels	$-23, -7$	$-21, -5$	$-19, -3$	$-17, -1$
Shift Levels	-7	-5	-3	-1
Levels	$1, 17$	$3, 19$	$5, 21$	$7, 23$
Shift Levels	1	3	5	7
Levels	$9, 25$	$11, 27$	$13, 29$	$15, 31$
Shift Levels	9	11	13	15

Corollary 2. *If Golay sequences of length ℓ exist, then a sliced orthogonal design for computer experiments $SOD(16\ell m, 8\ell, m)$ exists for any integer $m \geq 1$.*

The proof is provided in the Appendix. Table 2 summarises a variety of designs generated using the constructions developed in this paper. The first column specifies the design parameters, denoted as $SOD(Runs, Factors, Slices)$. The second column lists the sequences used, with their corresponding lengths given in the third column. The fourth column indicates the construction method applied. The final column describes the resulting level structure within a slice of each design.

To illustrate, consider the first row in Table 2: When using T-sequences of length $t = 1$, the resulting design is $SOD(8m, 4, m)$, that is, a design with m slices, each containing 8 runs and 4 factors.

None of the algorithms proposed in this paper requires the user to set the tuning parameters; all parameters are fully determined by the chosen construction method. The only input required from the user is the number of slices, m, which can be adjusted as needed. For example, suppose that a researcher wishes to construct an SOD with 12 factors and 48 runs, split across 2 slices. This can be achieved by applying Theorem 1 using T-sequences of length $t = 3$ and selecting $m = 2$ slices.

The proposed constructions offer several key advantages that make them suitable for a wide range of experimental contexts. For example, they provide full flexibility in the number of slices m, which can be arbitrarily increased. In addition, the structure of the design is adaptable: longer sequence lengths result in larger numbers of runs and factors per slice, enabling the designs to meet varying levels of complexity.

5. Generated designs and their properties

This section summarises the sliced orthogonal designs constructed using the proposed methodologies and evaluates them based on established criteria from the literature. Table 3 presents the key construction parameters and the sequences required for each design.

The first column, titled "Design in each slice $SOD(Runs, Factors)$", specifies the number of runs and factors within each slice. Since each design consists of m slices, and each slice is an orthogonal design for computer experiments (OD), the full SOD is formed by stacking these slices. The second column indicates the theorem or construction method used to generate each design. The third column reports the length of the sequences used: t for T-sequences, n for amicable disjoint sequences, and ℓ for Golay sequences.

The next two columns assess the aliasing with respect to quadratic terms. Specifically, $E(|q|)$ denotes the average absolute value of the alias matrix for quadratic effects (Equation 3.3), and $\max(|q|)$ gives the corresponding maximum absolute value (Equation 3.4). For all designs presented, the aliasing with respect to two-factor interactions is zero, i.e., $E(|t|) = 0$ and $\max(|t|) = 0$, as shown in Equations 3.1 and 3.2.

The final two columns evaluate space-filling properties using the distance criterion defined in Equation 3.5. The measure ϕ_{100}^{R} assesses the rectangular distance, while ϕ_{100}^{E} evaluates the Euclidean distance between the design points.

It is worth noting that multiple constructions can produce designs with the same parameters (runs, factors, levels), though small differences may occur in the evaluation criteria due to the different sequences and methods used for their construction.

6. Discussion

Significant efforts have been devoted to the design of computer experiments, yet the construction of sliced orthogonal designs for such experiments remains largely unexplored. Designs for computer experiments extend Latin hypercube designs by relaxing the requirement that the number of levels equal the number of runs. Although sliced Latin hypercube designs have been extensively studied in the literature, this paper presents a novel extension: the construction of sliced orthogonal designs for computer experiments (denoted SOD s). These designs are versatile and capable of accommodating both qualitative and quantitative factors. An SOD can exist in many cases where a sliced orthogonal Latin hypercube design (SOLHD) cannot.

The methodology developed in this study relies on the use of orthogonal square matrices and their foldovers to produce the desired designs. The construction is based on well-established complementary sequences, that is, T-sequences, Golay sequences, and disjoint amicable sequences, which play a central role in ensuring orthogonality. This approach ensures that the main effects are mutually orthogonal and uncorrelated with both the two-factor interaction terms and the quadratic effects.

Using these techniques, we have successfully constructed a variety of sliced orthogonal designs for computer experiments. Although our approach yields infinite families of such designs, the parameters are limited to values that are multiples of 4. Extending this method to allow for parameters not divisible by 4 remains an open direction for future research.

Acknowledgements

The authors thank the editor, associate editor, and reviewers for their valuable comments and suggestions, which have helped improve the quality of this article.

Appendix

Proof of Theorem 1

Proof. Let (T_1, T_2, T_3, T_4) be a set of T-sequences of length t. We now define, for each $i = 1, 2, \ldots, m$, the vectors:

$$A_1^i = (2i-1)T_1 + (2i+2m-1)T_2 + (2i+4m-1)T_3 + (2i+6m-1)T_4,$$

$$A_2^i = (1-2i-2m)T_1 + (2i-1)T_2 + (2i+6m-1)T_3 + (1-2i-4m)T_4,$$

$$A_3^i = (1-2i-4m)T_1 + (1-2i-6m)T_2 + (2i-1)T_3 + (2i+2m-1)T_4,$$

$$A_4^i = (1-2i-6m)T_1 + (2i+4m-1)T_2 + (1-2i-2m)T_3 + (2i-1)T_4.$$

From these vectors, define the circulant matrices $A_j^i = \mathrm{circ}(A_j^i)$ for $j = 1, 2, 3, 4$. Then, construct the $4t \times 4t$ matrix D_i using the Goethals–Seidel array:

$$D_i = \begin{pmatrix} A_1^i & A_2^i R_t & A_3^i R_t & A_4^i R_t \\ -A_2^i R_t & A_1^i & (A_4^i)^T R_t & -(A_3^i)^T R_t \\ -A_3^i R_t & -(A_4^i)^T R_t & A_1^i & (A_2^i)^T R_t \\ -A_4^i R_t & (A_3^i)^T R_t & -(A_2^i)^T R_t & A_1^i \end{pmatrix}.$$

where R_t is the back-diagonal identity matrix of size $t \times t$.

Now, by a result of Goethals and Seidel, D_i is orthogonal if the circulant matrices A_j^i satisfy:

$$\sum_{j=1}^{4} A_j^i (A_j^i)^T = \lambda I, \text{ where } \lambda = t(16i^2 + 48im - 16i + 56m^2 - 24m + 4).$$

These orthogonality conditions are met precisely because the sequences T_k are T-sequences and because the coefficients in the linear combinations used to construct A_j^i are carefully selected to cancel each other. From Cooper and Wallis (1972), it follows that the coefficient structure of each A_j^i satisfies the following:

$$\sum_{j=1}^{4} NPAF_{A_j^i}(0) = \lambda \quad \text{and} \quad \sum_{j=1}^{4} NPAF_{A_j^i}(s) = 0, \text{ for all } s = 1, 2, \ldots, t-1.$$

due to the properties of T-sequences.

This ensures that each D_i is orthogonal. Then, apply the foldover technique:

$$T_i = \begin{bmatrix} D_i^T & -D_i^T \end{bmatrix}^T, \quad i = 1, \ldots, m,$$

so that each T_i has dimensions $8t \times 4t$ and forms a slice of the final design.

The full sliced orthogonal design (SOD) is formed by stacking the transposes of the T_i:

$$X = \begin{bmatrix} T_1^T & T_2^T & \vdots & T_m^T \end{bmatrix}^T,$$

which gives an $8tm \times 4t$ design with m slices.

Finally, the levels are adjusted using the scheme in equation (4.1), preserving orthogonality between the main effects and their independence from second-order effects (interactions and quadratics) within each slice.

$\square$

Proof of Theorem 2:

Proof. For generating our construction from Golay sequences A_1 and A_2 of length ℓ. From these Golay sequences, we define three extended vectors:

$$X_1 = \{1, \overbrace{0, \ldots, 0}^{2\ell}\},\ X_2 = \{0, A_1, \overbrace{0, \ldots, 0}^{\ell}\}, \text{ and } X_3 = \{\overbrace{0, \ldots, 0}^{\ell+1}, A_2\}$$

Using the vectors above, we define four vectors $A_1^i, A_2^i, A_3^i, A_4^i$ for each i as follows:

$$A_1^i = (2i-1)X_1 + (2i+2m-1)X_2 + (2i+4m-1)X_3$$

$$A_2^i = (1-2i-2m)X_1 + (2i-1)X_2 + (2i+6m-1)X_3$$

$$A_3^i = (1-2i-4m)X_1 + (1-2i-6m)X_2 + (2i-1)X_3$$

$$A_4^i = (1-2i-6m)X_1 + (2i+4m-1)X_2 + (1-2i-2m)X_3.$$

These vectors have zero $NPAF$ and can be used to construct circulant matrices, which can then be inserted into the Goethals–Seidel array (GS-array) of the following form:

$$D_i = \begin{pmatrix} A_1^i & A_2^i R_n & A_3^i R_n & A_4^i R_n \\ -A_2^i R_n & A_1^i & (A_4^i)^T R_n & -(A_3^i)^T R_n \\ -A_3^i R_n & -(A_4^i)^T R_n & A_1^i & (A_2^i)^T R_n \\ -A_4^i R_n & (A_3^i)^T R_n & -(A_2^i)^T R_n & A_1^i \end{pmatrix}.$$

where R_n denotes the reverse identity matrix of order $n = 2\ell + 1$.

Each matrix D_i, for $i = 1, 2, \ldots, m$, is orthogonal.

To generate the design slices, we apply a fold-over technique to each D_i, resulting in:$T_i = [D_i^T, -D_i^T]^T, \quad i = 1, 2, \ldots, m$ where each T_i is orthogonal and considered as a slice in the overall design. It is important to note that each slice in our construction is formed using an orthogonal matrix derived from a Goethals-Seidel array, which ensures the orthogonality of the individual slices. Consequently, bringing these slices together results in a design that maintains orthogonality overall and throughout all layers.

The final Sliced Orthogonal Design SOD consists of m slices, $8nm$ runs, and $4n$ factors, where $n = 2\ell + 1$. It is formally represented by:$X = [T_1^T, T_2^T, \ldots, T_m^T]^T$. We then apply the adjustment procedure for the levels in each slice as described in Equation 4.1. $\square$

Proof of Theorem 3:

Proof. The construction of the desired design begins with the selection of two disjoint amicable sequences, X_1, X_2, each consisting of elements from the set $\{0, \pm 1\}$, with length n. These sequences are used to generate a set of eight vectors for each index i, given by:

$$
\begin{array}{ll}
A^i_1 = (16i-15)X_1 + (16i-13)X_2 & A^i_5 = (16i-7)X_1 + (16i-5)X_2 \\
A^i_2 = (13-16i)X_1 + (16i-15)X_2 & A^i_6 = (5-16i)X_1 + (16i-7)X_2 \\
A^i_3 = (16i-11)X_1 + (16i-9)X_2 & A^i_7 = (16i-3)X_1 + (16i-1)X_2 \\
A^i_4 = (9-16i)X_1 + (16i-11)X_2 & A^i_8 = (1-16i)X_1 + (16i-3)X_2
\end{array}
$$

Using the properties of disjoint amicable sequences it is easy to show that these vectors have zero NPAF and can be used to construct circulant matrices, which can then be inserted into the Kharaghani array. The circulant matrix D_i for each index i is given by: D_i=

$$
\begin{pmatrix}
A^i_1 & A^i_2 & A^i_4R_n & A^i_3R_n & A^i_6R_n & A^i_5R_n & A^i_8R_n & A^i_7R_n \\
-A^i_2 & A^i_1 & A^i_3R_n & -A^i_4R_n & A^i_5R_n & -A^i_6R_n & A^i_7R_n & -A^i_8R_n \\
-A^i_4R_n & -A^i_3R_n & A^i_1 & A^i_2 & -(A^i_8)^TR_n & (A^i)^T_7R_n & (A^i_6)^TR_n & -(A^i_5)^TR_n \\
-A^i_3R_n & A^i_4R_n & -A^i_2 & A^i_1 & (A^i_7)^TR_n & (A^i_8)^TR_n & -(A^i_5)^TR_n & -(A^i_6)^TR_n \\
-A^i_6R_n & -A^i_5R_n & (A^i_8)^TR_n & -(A^i_7)^TR_n & A^i_1 & A^i_2 & -(A^i_4)^TR_n & (A^i_3)^TR_n \\
-A^i_5R_n & A^i_6R_n & -(A^i_7)^TR_n & -(A^i_8)^TR_n & -A^i_2 & A^i_1 & (A^i_3)^TR_n & (A^i_4)^TR_n \\
-A^i_8R_n & -A^i_7R_n & -(A^i_6)^TR_n & (A^i_5)^TR_n & (A^i_4)^TR_n & -(A^i_3)^TR_n & A^i_1 & A^i_2 \\
-A^i_7R_n & A^i_8R_n & (A^i_5)^TR_n & (A^i_6)^TR_n & -(A^i_3)^TR_n & -(A^i_4)^TR_n & -A^i_2 & A^i_1
\end{pmatrix}.
$$

Each matrix D_i is orthogonal and has dimension $8n \times 8n$, for $i = 1, 2, \ldots, m$.

We then apply the foldover technique to each D_i, producing

$$T_i = \begin{bmatrix} D_i^T & -D_i^T \end{bmatrix}^T, \quad i = 1, 2, \ldots, m.$$

This construction ensures that each T_i forms an orthogonal slice derived from a Kharaghani array. Since each slice is orthogonal, stacking these slices preserves orthogonality across the entire design.

The final Sliced Orthogonal Design (SOD) is then constructed by stacking the slices together:

$$X = \begin{bmatrix} T_1^T & T_2^T & \cdots & T_m^T \end{bmatrix}^T.$$

This design consists of m slices, with a total of $16nm$ runs and $8n$ factors. The levels within each slice are adjusted according to the scheme described in equation (4.4). □

Proof of Colloary 2:

Proof. The proof is straightforward like the proof of Theorem 3, where X_1 and X_2, as needed in Theorem 3, can be constructed as follows:

$$X_1 = \frac{A+B}{2}, \quad X_2 = \frac{A-B}{2}$$

where A, B are Golay sequences of length ℓ. X_1 and X_2 are two disjoint amicable sequences, each consisting of elements from the set $\{0, \pm 1\}$, with length ℓ. The result follows from Theorem 3. □

Full Matrices for Example 3

This appendix provides the complete Goethals–Seidel matrices D_1 and D_2 used in Example 3 to construct the sliced orthogonal design $SOD(80, 20, 2)$.

Matrix D_1

$$
\begin{pmatrix}
1 & 5 & 5 & 9 & -9 & -13 & 13 & 1 & 1 & -5 & -1 & 1 & -13 & -13 & -9 & 5 & -5 & 9 & 9 & -13 \\
-9 & 1 & 5 & 5 & 9 & 13 & 1 & 1 & -5 & -13 & 1 & -13 & -13 & -9 & -1 & -5 & 9 & 9 & -13 & 5 \\
9 & -9 & 1 & 5 & 5 & 1 & 1 & -5 & -13 & 13 & -13 & -13 & -9 & -1 & 1 & 9 & 9 & -13 & 5 & -5 \\
5 & 9 & -9 & 1 & 5 & 1 & -5 & -13 & 13 & 1 & -13 & -9 & -1 & 1 & -13 & 9 & -13 & 5 & -5 & 9 \\
5 & 5 & 9 & -9 & 1 & -5 & -13 & 13 & 1 & 1 & -9 & -1 & 1 & -13 & -13 & -13 & 5 & -5 & 9 & 9 \\
13 & -13 & -1 & -1 & 5 & 1 & 5 & 5 & 9 & -9 & -9 & -9 & 5 & -5 & 13 & -13 & -13 & 1 & -1 & -9 \\
-13 & -1 & -1 & 5 & 13 & -9 & 1 & 5 & 5 & 9 & -9 & 5 & -5 & 13 & -9 & -13 & 1 & -1 & -9 & -13 \\
-1 & -1 & 5 & 13 & -13 & 9 & -9 & 1 & 5 & 5 & 5 & -5 & 13 & -9 & -9 & 1 & -1 & -9 & -13 & -13 \\
-1 & 5 & 13 & -13 & -1 & 5 & 9 & -9 & 1 & 5 & -5 & 13 & -9 & -9 & 5 & -1 & -9 & -13 & -13 & 1 \\
5 & 13 & -13 & -1 & -1 & 5 & 5 & 9 & -9 & 1 & 13 & -9 & -9 & 5 & -5 & -9 & -13 & -13 & 1 & -1 \\
1 & -1 & 13 & 13 & 9 & 9 & 9 & -5 & 5 & -13 & 1 & 5 & 5 & 9 & -9 & -1 & -1 & -13 & 13 & 5 \\
-1 & 13 & 13 & 9 & 1 & 9 & -5 & 5 & -13 & 9 & -9 & 1 & 5 & 5 & 9 & -1 & -13 & 13 & 5 & -1 \\
13 & 13 & 9 & 1 & -1 & -5 & 5 & -13 & 9 & 9 & 9 & -9 & 1 & 5 & 5 & -13 & 13 & 5 & -1 & -1 \\
13 & 9 & 1 & -1 & 13 & 5 & -13 & 9 & 9 & -5 & 5 & 9 & -9 & 1 & 5 & 13 & 5 & -1 & -1 & -13 \\
9 & 1 & -1 & 13 & 13 & -13 & 9 & 9 & -5 & 5 & 5 & 5 & 9 & -9 & 1 & 5 & -1 & -1 & -13 & 13 \\
-5 & 5 & -9 & -9 & 13 & 13 & 13 & -1 & 1 & 9 & 1 & 1 & 13 & -13 & -5 & 1 & 5 & 5 & 9 & -9 \\
5 & -9 & -9 & 13 & -5 & 13 & -1 & 1 & 9 & 13 & 1 & 13 & -13 & -5 & 1 & -9 & 1 & 5 & 5 & 9 \\
-9 & -9 & 13 & -5 & 5 & -1 & 1 & 9 & 13 & 13 & 13 & -13 & -5 & 1 & 1 & 9 & -9 & 1 & 5 & 5 \\
-9 & 13 & -5 & 5 & -9 & 1 & 9 & 13 & 13 & -1 & -13 & -5 & 1 & 1 & 13 & 5 & 9 & -9 & 1 & 5 \\
13 & -5 & 5 & -9 & -9 & 9 & 13 & 13 & -1 & 1 & -5 & 1 & 1 & 13 & -13 & 5 & 5 & 9 & -9 & 1
\end{pmatrix}
$$

Matrix D_2

$$\begin{pmatrix}
3 & 7 & 7 & 11 & -11 & -15 & 15 & 3 & 3 & -7 & -3 & 3 & -15 & -15 & -11 & 7 & -7 & 11 & 11 & -15 \\
-11 & 3 & 7 & 7 & 11 & 15 & 3 & 3 & -7 & -15 & 3 & -15 & -15 & -11 & -3 & -7 & 11 & 11 & -15 & 7 \\
11 & -11 & 3 & 7 & 7 & 3 & 3 & -7 & -15 & 15 & -15 & -15 & -11 & -3 & 3 & 11 & 11 & -15 & 7 & -7 \\
7 & 11 & -11 & 3 & 7 & 3 & -7 & -15 & 15 & 3 & -15 & -11 & -3 & 3 & -15 & 11 & -15 & 7 & -7 & 11 \\
7 & 7 & 11 & -11 & 3 & -7 & -15 & 15 & 3 & 3 & -11 & -3 & 3 & -15 & -15 & -15 & 7 & -7 & 11 & 11 \\
15 & -15 & -3 & -3 & 7 & 3 & 7 & 7 & 11 & -11 & -11 & -11 & 7 & -7 & 15 & -15 & -15 & 3 & -3 & -11 \\
-15 & -3 & -3 & 7 & 15 & -11 & 3 & 7 & 7 & 11 & -11 & 7 & -7 & 15 & -11 & -15 & 3 & -3 & -11 & -15 \\
-3 & -3 & 7 & 15 & -15 & 11 & -11 & 3 & 7 & 7 & 7 & -7 & 15 & -11 & -11 & 3 & -3 & -11 & -15 & -15 \\
-3 & 7 & 15 & -15 & -3 & 7 & 11 & -11 & 3 & 7 & -7 & 15 & -11 & -11 & 7 & -3 & -11 & -15 & -15 & 3 \\
7 & 15 & -15 & -3 & -3 & 7 & 7 & 11 & -11 & 3 & 15 & -11 & -11 & 7 & -7 & -11 & -15 & -15 & 3 & -3 \\
3 & -3 & 15 & 15 & 11 & 11 & 11 & -7 & 7 & -15 & 3 & 7 & 7 & 11 & -11 & -3 & -3 & -15 & 15 & 7 \\
-3 & 15 & 15 & 11 & 3 & 11 & -7 & 7 & -15 & 11 & -11 & 3 & 7 & 7 & 11 & -3 & -15 & 15 & 7 & -3 \\
15 & 15 & 11 & 3 & -3 & -7 & 7 & -15 & 11 & 11 & 11 & -11 & 3 & 7 & 7 & -15 & 15 & 7 & -3 & -3 \\
15 & 11 & 3 & -3 & 15 & 7 & -15 & 11 & 11 & -7 & 7 & 11 & -11 & 3 & 7 & 15 & 7 & -3 & -3 & -15 \\
11 & 3 & -3 & 15 & 15 & -15 & 11 & 11 & -7 & 7 & 7 & 7 & 11 & -11 & 3 & 7 & -3 & -3 & -15 & 15 \\
-7 & 7 & -11 & -11 & 15 & 15 & 15 & -3 & 3 & 11 & 3 & 3 & 15 & -15 & -7 & 3 & 7 & 7 & 11 & -11 \\
7 & -11 & -11 & 15 & -7 & 15 & -3 & 3 & 11 & 15 & 3 & 15 & -15 & -7 & 3 & -11 & 3 & 7 & 7 & 11 \\
-11 & -11 & 15 & -7 & 7 & -3 & 3 & 11 & 15 & 15 & 15 & -15 & -7 & 3 & 3 & 11 & -11 & 3 & 7 & 7 \\
-11 & 15 & -7 & 7 & -11 & 3 & 11 & 15 & 15 & -3 & -15 & -7 & 3 & 3 & 15 & 7 & 11 & -11 & 3 & 7 \\
15 & -7 & 7 & -11 & -11 & 11 & 15 & 15 & -3 & 3 & -7 & 3 & 3 & 15 & -15 & 7 & 7 & 11 & -11 & 3
\end{pmatrix}$$

Each D_i is a 20×20 orthogonal matrix used to construct the slices T_1 and T_2 in $SOD(80, 20, 2)$.

References

Alhelali, O. A., S. Georgiou, C. Koukouvinos, and S. Stylianou (2023). Orthogonal designs for computer experiments constructed from sequences with zero autocorrelation. *Applied*

Numerical Mathematics.

Bingham, D., R. R. Sitter, and B. Tang (2009). Orthogonal and nearly orthogonal designs for computer experiments. *Biometrika 96*(1), 51–65.

Cao, R.-Y. and M.-Q. Liu (2015). Construction of second-order orthogonal sliced latin hypercube designs. *Journal of Complexity 31*(5), 762–772.

Chen, P.-H. A., T. J. Santner, and A. M. Dean (2018). Sequential pareto minimization of physical systems using calibrated computer simulators. *Statistica Sinica*, 671–692.

Cohen, G., D. Rubie, J. Seberry, C. Koukouvinos, S. Kounias, and M. Yamada (1989). A survey of base sequences, disjoint complementary sequences and od (4t; t, t, t, t). *J. Comb. Math. Comb. Comp*, 69–104.

Colbourn, C. J. and J. H. Dinitz (2006). *Handbook of Combinatorial Designs, Second Edition (Discrete Mathematics and Its Applications)*. Chapman & Hall/CRC.

Cooper, J. and J. S. Wallis (1972). A construction for hadamard arrays. *Bulletin of the Australian Mathematical Society 7*(2), 269–278.

Djokovic, D. Z. (2010a). A new yang number and consequences. *Designs, Codes and Cryptography 54*, 201–204.

Djokovic, D. Z. (2010b). Small orders of hadamard matrices and base sequences. *arXiv preprint arXiv:1008.2043*.

Fang, K.-T., R. Li, and A. Sudjianto (2005). *Design and modeling for computer experiments.*

CRC press.

Fang, K.-T., D. K. Lin, and H. Qin (2003). A note on optimal foldover design. *Statistics & probability letters 62*(3), 245–250.

Georgiou, S., C. Koukouvinos, and J. Seberry (2002). Short amicable sets. *International Journal of Applied Mathematics*.

Georgiou, S. D. (2011). Orthogonal designs for computer experiments. *Journal of Statistical Planning and Inference 141*(4), 1519–1525.

Georgiou, S. D. and I. Efthimiou (2014). Some classes of orthogonal latin hypercube designs. *Statistica Sinica 24*(1), 101–120.

Guo, B., X.-R. Li, M.-Q. Liu, and X. Yang (2023). Construction of orthogonal general sliced latin hypercube designs. *Statistical Papers 64*(3), 987–1014.

He, Y. and B. Tang (2013). Strong orthogonal arrays and associated latin hypercubes for computer experiments. *Biometrika 100*(1), 254–260.

Huang, H., J.-F. Yang, and M.-Q. Liu (2014). Construction of sliced (nearly) orthogonal latin hypercube designs. *Journal of Complexity 30*(3), 355–365.

Huang, H., H. Yu, M.-Q. Liu, and D. Wu (2021). Construction of uniform designs and complex-structured uniform designs via partitionable t-designs. *Statistica Sinica 31*(4), 1689–1706.

Joseph, V. R. and Y. Hung (2008). Orthogonal-maximin latin hypercube designs. *Statistica Sinica*, 171–186.

Kharaghani, H. (2000). Arrays for orthogonal designs. *Journal of Combinatorial Designs 8*(3), 166–173.

Kumar, A. A., B. N. Mandal, R. Parsad, S. Dash, and M. Kumar (2024). On construction of sliced orthogonal latin hypercube designs. *Journal of Statistical Theory and Practice 18*(4), 52.

Li, W., M.-Q. Liu, and B. Tang (2021). A method of constructing maximin distance designs. *Biometrika 108*(4), 845–855.

Lin, C. D. and B. Tang (2022). Latin hypercubes and space-filling designs. *arXiv preprint arXiv:2203.06334*.

McKay, M. D., R. J. Beckman, and W. J. Conover (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics 42*(1), 55–61.

Morris, M. D. and T. J. Mitchell (1995). Exploratory designs for computational experiments. *Journal of statistical planning and inference 43*(3), 381–402.

Qian, P. Z. (2012). Sliced latin hypercube designs. *Journal of the American Statistical Association 107*(497), 393–399.

Santner, T. J., B. J. Williams, W. I. Notz, and B. J. Williams (2003). *The design and analysis of computer experiments*, Volume 1. Springer.

Seberry, J. (2017, 01). *Orthogonal Designs: Hadamard Matrices, Quadratic Forms and Algebras.*

Cham: Springer.

Seberry, J. and M. Yamada (1992). Hadamard matrices, sequences and block designs. In J. H. Dinitz and D. R. Stinson (Eds.), *Contemporary Design Theory – A Collection of Surveys*, pp. 431–560. New York: John Wiley and Sons.

Shi, C. and B. Tang (2020). Construction results for strong orthogonal arrays of strength three. *Bernoulli 26*(1), 418 – 431.

Steinberg, D. M. and D. K. Lin (2006). A construction method for orthogonal latin hypercube designs. *Biometrika 93*(2), 279–288.

Sun, F. and B. Tang (2017). A general rotation method for orthogonal latin hypercubes. *Biometrika 104*(2), 465–472.

Tang, B. (1993). Orthogonal array-based latin hypercubes. *Journal of the American statistical association 88*(424), 1392–1397.

Wang, C., J. Yang, and M.-Q. Liu (2022). Construction of strong group-orthogonal arrays. *Statistica Sinica 32*(3), 1–19.

Wang, L., F. Sun, D. K. Lin, and M.-Q. Liu (2018). Construction of orthogonal symmetric latin hypercube designs. *Statistica Sinica 28*(3), 1503–1520.

Wang, L., Q. Xiao, and H. Xu (2018). Optimal maximin L_1-distance Latin hypercube designs based on good lattice point designs. *The Annals of Statistics 46*(6B), 3741 – 3766.

Wang, X.-L., Y.-N. Zhao, J.-F. Yang, and M.-Q. Liu (2017). Construction of (nearly) orthogonal sliced latin hypercube designs. *Statistics & Probability Letters 125*, 174–180.

Xiao, Q. and H. Xu (2018). Construction of maximin distance designs via level permutation and expansion. *Statistica Sinica 28*(3), 1395–1414.

Yang, J., H. Chen, D. K. Lin, and M.-Q. Liu (2016). Construction of sliced maximin-orthogonal latin hypercube designs. *Statistica Sinica*, 589–603.

Yang, J.-F., C. D. Lin, P. Z. Qian, and D. K. Lin (2013). Construction of sliced orthogonal latin hypercube designs. *Statistica Sinica*, 1117–1130.

Zhou, Y. and B. Tang (2019). Column-orthogonal strong orthogonal arrays of strength two plus and three minus. *Biometrika 106*(4), 997–1004.

Omar A. Alhelali

Mathematical Sciences, School of Science, RMIT University, Melbourne, VIC 3001, Australia.

Dept. of Mathematics, Applied College, King Khalid University, Muhayil Asir, Saudi Arabia.

E-mail: s3609226@student.rmit.edu.au

S.D. Georgiou

Mathematical Sciences, School of Science, RMIT University, Melbourne, VIC 3001, Australia.

E-mail: stelios.georgiou@rmit.edu.au

S. Stylianou

Mathematical Sciences, School of Science, RMIT University, Melbourne, VIC 3001, Australia.

E-mail: stella.stylianou@rmit.edu.au

Table 2: Some Sliced orthogonal designs ($SODs$) via the proposed methods

.

D(Runs, Factors)	Sequences	Length	Method	Levels
$SOD(8m, 4)$	T-sequences	$t = 1$	Theorem 1	8
$SOD(16m, 8)$	Disjoint Amicable sequences	$n = 1$	Theorem 3	16
$SOD(24m, 12)$	T-sequences	$t = 3$	Theorem 1	8
$SOD(32m, 16)$	Disjoint Amicable sequences	$n = 2$	Theorem 3	16
	Golay sequences	$l = 2$	Corollary 2	
$SOD(40m, 20)$	T-sequences	$t = 5$	Theorem 1	8
	Golay sequences	$l = 2$	Theorem 2	
$SOD(56m, 28)$	T-sequences	$t = 7$	Theorem 1	8
$SOD(64m, 32)$	Disjoint Amicable sequences	$n = 4$	Theorem 3	16
	Golay sequences	$l = 4$	Corollary 2	
$SOD(72m, 36)$	T-sequences	$t = 9$	Theorem 1	8
	Golay sequences	$l = 4$	Theorem 2	
$SOD(88m, 44)$	T-sequences	$t = 11$	Theorem 1	8
$SOD(96m, 48)$	Disjoint Amicable sequences	$n = 6$	Theorem 3	17
$SOD(104m, 52)$	T-sequences	$t = 13$	Theorem 1	8
$SOD(112m, 56)$	Disjoint Amicable sequences	$n = 7$	Theorem 3	17

Table 2: (cont.))

D(Runs, Factors)	**Sequences**	**Length**	**Method**	**Levels**
$SOD(120m, 60)$	T-sequences	$t = 15$	Theorem 1	8
$SOD(128m, 64)$	Disjoint Amicable sequences	$n = 8$	Theorem 3	16
	Golay sequences	$l = 8$	Corollary 2	
$SOD(136m, 68)$	T-sequences	$t = 17$	Theorem 1	8
	Golay sequences	$l = 8$	Theorem 2	
$SOD(152m, 76)$	T-sequences	$t = 19$	Theorem 1	8
$SOD(160m, 80)$	Disjoint Amicable sequences	$n = 10$	Theorem 3	17
	Golay sequences	$l = 10$	Corollary 2	16
$SOD(168m, 84)$	T-sequences	$t = 31$	Theorem 1	8
	Golay sequences	$l = 10$	Theorem 2	
$SOD(184m, 92)$	T-sequences	$t = 23$	Theorem 1	8
$SOD(200m, 100)$	T-sequences	$t = 25$	Theorem 1	8
$SOD(256m, 128)$	Golay sequences	$l = 16$	Corollary 2	16
$SOD(264m, 132)$	Golay sequences	$l = 16$	Theorem 2	8
$SOD(320m, 160)$	Golay sequences	$l = 20$	Corollary 2	16
$SOD(328m, 164)$	Golay sequences	$l = 20$	Theorem 2	8

Table 3: Designs constructed by our Theorems. These designs have $E(t) = max(t) = 0$

Design in each Slice $SOD(Runs, Factors)$	**Construction** $(Method)$	**Length**	$E(q)$	$max(q)$	ϕ_{100}^{R}	ϕ_{100}^{E}
$SOD(8,4)$	Theorem 1	$t=1$	0.0857	0.4286	0.3574	0.5575
$SOD(16,8)$	Theorem 3	$n=1$	0.0443	0.3984	0.2056	0.4151
$SOD(24,12)$	Theorem 1	$t=3$	0.033	0.4286	0.1365	0.3297
$SOD(32,16)$	Theorem 3	$n=2$	0.0222	0.3778	0.1213	0.3059
$SOD(32,16)$	Corollary 2	$l=2$	0.0222	0.3778	0.1090	0.3059
$SOD(40,20)$	Theorem 1	$t=5$	0.0204	0.4286	0.0845	0.2581
$SOD(40,20)$	Theorem 2	$l=2$	0.0204	0.4286	0.0813	0.2581
$SOD(56,28)$	Theorem 1	$t=7$	0.0148	0.4286	0.0572	0.2196
$SOD(64,32)$	Theorem 3	$n=4$	0.0114	0.3778	0.0618	0.2194
$SOD(64,32)$	Corollary 2	$l=4$	0.0114	0.3778	0.0579	0.2194
$SOD(72,36)$	Theorem 1	$t=9$	0.0116	0.4286	0.0437	0.1947
$SOD(72,36)$	Theorem 2	$l=4$	0.0116	0.4286	0.0471	0.1947
$SOD(88,44)$	Theorem 1	$t=11$	0.0095	0.4286	0.0394	0.1768
$SOD(96,48)$	Theorem 3	$n=6$	0.0051	0.2519	0.0613	0.2212
$SOD(104,52)$	Theorem 1	$t=13$	0.0081	0.4286	0.0310	0.1632
$SOD(112,56)$	Theorem 3	$n=7$	0.004	0.2277	0.0524	0.2161
$SOD(120,60)$	Theorem 1	$t=15$	0.0070	0.4286	0.0269	0.1523

Table 3: (cont.))

Design in each Slice $SOD(Runs, Factors)$	**Construction** $(Method)$	**Length**	$E(q)$	$max(q)$	ϕ_{100}^{R}	ϕ_{100}^{E}
$SOD(128, 64)$	Theorem 3	$n = 8$	0.0058	0.3778	0.0308	0.1573
$SOD(128, 64)$	Corollary 2	$l = 8$	0.0058	0.3778	0.0300	0.1573
$SOD(136, 68)$	Theorem 1	$t = 17$	0.0062	0.4286	0.0258	0.1435
$SOD(136, 68)$	Theorem 2	$l = 8$	0.0062	0.4286	0.0258	0.1435
$SOD(152, 76)$	Theorem 1	$t = 19$	0.0056	0.4286	0.0225	0.1360
$SOD(160, 80)$	Theorem 3	$n = 10$	0.0042	0.34	0.0252	0.1490
$SOD(160, 80)$	Corollary 2	$l = 10$	0.0047	0.3778	0.0238	0.1414
$SOD(168, 84)$	Theorem 1	$t = 21$	0.0050	0.4286	0.0211	0.1300
$SOD(168, 84)$	Theorem 2	$l = 10$	0.0050	0.4286	0.0211	0.1297
$SOD(184, 92)$	Theorem 1	$t = 23$	0.0046	0.4286	0.0184	0.1241
$SOD(200, 100)$	Theorem 1	$t = 25$	0.0042	0.4286	0.0165	0.1192
$SOD(216, 108)$	Theorem 1	$t = 27$	0.0039	0.4286	0.0163	0.1149
$SOD(232, 116)$	Theorem 1	$t = 29$	0.0037	0.4286	0.0143	0.1110
$SOD(248, 124)$	Theorem 1	$t = 31$	0.0034	0.4286	0.0142	0.1075
$SOD(256, 128)$	Corollary 2	$l = 16$	0.0029	0.3778	0.0154	0.1128
$SOD(264, 132)$	Theorem 2	$l = 16$	0.0032	0.4286	0.0137	0.1044
$SOD(264, 132)$	Theorem 1	$t = 33$	0.0032	0.4286	0.0118	0.1044
$SOD(280, 140)$	Theorem 1	$t = 35$	0.0030	0.4286	0.0118	0.1015

Table 3: (cont.))

Design in each Slice $SOD(Runs, Factors)$	**Construction** $(Method)$	**Length**	$E(q)$	$max(q)$	ϕ_{100}^{R}	ϕ_{100}^{E}
$SOD(296, 148)$	Theorem 1	$t = 37$	0.0029	0.4286	0.0112	0.0988
$SOD(313, 156)$	Theorem 1	$t = 39$	0.0027	0.4286	0.0105	0.0963
$SOD(320, 160)$	Corollary 2	$l = 20$	0.0023	0.3778	0.0122	0.1014
$SOD(328, 164)$	Theorem 2	$l = 20$	0.0026	0.4286	0.0111	0.0940