Available online at www.sciencedirect.com

SCIENCE DIRECT®

Statistics & Probability Letters 75 (2005) 169–178

STATISTICS & PROBABILITY LETTERS

www.elsevier.com/locate/stapro

Some new orthogonal arrays OA($4r$, $r^1 2^p$, 2)

Warren F. Kuhfeld[a,*], Chung-yi Suen[b]

[a]*SAS Institute Inc., S3018 SAS Campus Drive, Cary, NC 27513-2414, USA*
[b]*Cleveland State University, Department of Mathematics, 2121 Euclid Avenue, RT 1515, Cleveland, OH 44115-2214, USA*

Received 4 November 2004; received in revised form 12 May 2005; accepted 21 May 2005
Available online 5 July 2005

Abstract

We developed an algorithm to search for new orthogonal arrays, OA($4r$, $r^1 2^p$, 2), for odd r. With it, we found new orthogonal arrays with $4r = 36$ through 124 runs. Many other new arrays can be obtained from these new arrays.
© 2005 Elsevier B.V. All rights reserved.

MSC: primary 62K15

Keywords: Juxtaposition; Orthogonal array; Resolvable; SAS/IML

1. Introduction

Let OA($n, a^p b^q, \ldots, 2$) denote a strength 2 orthogonal array with n runs or rows and $p + q + \cdots$ columns or factors, p with a levels, q with b levels, and so on. It is well known that strength 2 orthogonal arrays are useful plans for main-effects experimental designs (Rao, 1947; Hedayat et al., 1999; and many others).

We developed an algorithm to search for the maximum number of two-level factors, p, in orthogonal arrays of the form OA($4r$, $r^1 2^p$, 2). Small examples from this family include OA(12, $3^1 2^4$, 2), OA(20, $5^1 2^8$, 2), and OA(28, $7^1 2^{12}$, 2). If the r-level column of the

*Corresponding author. Tel.: +1 919 5317922; fax: +1 919 6774444.
E-mail address: warren.kuhfeld@sas.com (W.F. Kuhfeld).

0167-7152/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.spl.2005.05.006

OA($4r$, $r^1 2^p$, 2) is deleted, then it becomes a resolvable OA($4r$, 2^p, 2) which can be partitioned into r strength 1 orthogonal arrays. Suen (1989) considered the construction of OA($4r$, $r^1 2^p$, 2) (or resolvable OA($4r$, 2^p, 2)). It is known that if $r = 1, 2$ or a multiple of 4 (i.e. r is a Hadamard number), then p can reach its upper bound $3r$. If r is even but not a multiple of 4, then an OA($4r$, $r^1 2^{2r+2}$, 2) (i.e. $p = 2r + 2$) can be constructed. When $r > 1$ is an odd number, a theoretical construction is not known. We can only find this type of orthogonal array with computerized searches.

Our search algorithm is based on the work of Suen (1989). Suen showed that OA($4r$, $r^1 2^p$, 2) can be constructed from an $r \times p$ matrix with entries: $x, y, z, -x, -y, -z$ where $x' = (1\,1 - 1 - 1)$, $y' = (1 - 1\,1 - 1)$, and $z' = (1 - 1 - 1\,1)$. In the first two examples with 12 and 20 runs, Suen showed the maximum number of two-level factors is 4 and 8, respectively. For all larger cases, the maximum p is unknown. With our algorithm and modern computers, we were able to search longer and faster and hence find designs that he could not find. Our results show the maximum p that we found empirically for each case, not a combinatorially proven maximum.

2. Search algorithm

Our algorithm is straightforward, and it is shown in its entirety in the appendix. The implementation is in SAS/IML® (SAS, 2003). Other approaches, such as compiled C, would run faster, but an IML implementation is both convenient and compact. Furthermore, what we lose in efficiency we can gain by running on multiple computers. The algorithm starts by initializing column 1 with r copies of x. Then the algorithm seeks a second orthogonal column given the first, then a third orthogonal column given the first two, and so on. The algorithm is entirely sequential; no attempts are made to go back and change a previously found column. When the algorithm gives up without finding the next column, it goes back and starts over again beginning with column two. This outer loop is similar to the approach Xu (2002) used to find new orthogonal arrays and Kuhfeld (2005b) used to find new difference schemes.

Each search for a new column consists of repeatedly running several steps, each time based on a different random start. First, a $4r \times 1$ vector consisting of r subvectors chosen randomly from $(x, y, z, -x, -y, -z)$ is generated, then the scalar products are evaluated with the previous columns. Care is taken to minimize the size of the scalar products computed at each step and update rather than recalculate results. The algorithm tries replacing each of the r entries in turn with each element of $(x, y, z, -x, -y, -z)$. Replacements that move the scalar products closer to zero are performed until either an orthogonal column is found or the algorithm gives up and tries a new random start. For large problems and higher-order columns, it may take many random starts to find the next orthogonal column, so it is important to not give up too early. For example, the last three columns of OA$(116, 29^1 2^{23}, 2)$ were found after (21, 26, 90)-million tries.

The maximum number of random starts for each column varies by column number. Throughout the course of our searches, we made many changes to these numbers. This implementation generates maxima of 2.3^j, where j is the index of the column being sought. For example, the maximum number of random starts for columns 5, 10, 15, and 20 are 65, 4143, 266,636, and 17,161,559, respectively. The maximum number of searches is set to 100 million for column 23 and beyond. The algorithm will start over at column two very quickly if it has trouble finding any of the first few columns. However, the farther it gets, the longer it works on each new