U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

NISTIR 5121

Computer Systems Laboratory

Building Hadamard Matrices in Steps of 4 to Order 200

Nathalie Drouin

U. S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards and Technology
Gaithersburg, MD 20899

April 1993

CMRF

COMPUTER MEASUREMENT
RESEARCH FACILITY
FOR HIGH PERFORMANCE
PARALLEL COMPUTATION

NISTIR 5121

Building Hadamard Matrices in Steps of 4 to Order 200

Nathalie Drouin

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards
and Technology
Computer Systems Laboratory
Gaithersburg, MD 20899

April 1993

U.S. DEPARTMENT OF COMMERCE
Ronald H. Brown, Secretary

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
Raymond G. Kammer, Acting Director

TABLE OF CONTENTS

Building Hadamard Matrices in Steps of 4 to Order 200

Nathalie DROUIN

Based on methods of construction described in 1978 by I.A.Hedayat and D.W.Wallis, programs described herein allow one to build Hadamard matrices of order up to 200, in steps of 4. These matrices are to be used to generate statistical plans of analysis for the "Synthetic Perturbation Tuning" technique of program sensitivity analysis.

Key words : Design of Experiments; Fractional Factorial Design; Hadamard Matrices; Synthetic Perturbation Tuning.

1. Introduction

The use of factorial designs has become widely accepted as an efficient way for carrying out experiments involving many different factors. However, the number of measurements required may be large and in some cases prohibitive. Such is the case for the motivating application for this study, *Synthetic Perturbation Tuning (SPT)* [LSK92]. SPT is a method for determining computer program performance sensitivities via fractional designs. Because computer programs can be short or long, SPT can present a set of factors ranging from few to many.

Several statisticians have recently devoted attention to the problem of conducting factorial experiments requiring a reduced set of measurements. Experimental designs of this type are commonly referred to as fractional factorial designs or fractional replicates. Hadamard matrices are intimately connected to factorial experiments in which each factor is at two levels. Furthermore, their application to types of design such as optimal resolution 3 designs has been discovered very recently. Among other characteristics, it has been proved that the existence of a Hadamard matrix of order *4t* implies the existence of an orthogonal fractional factorial design of resolution 3 for *(4t-1)* factors each at two levels. This is currently of interest to us because other methods do not provide so fine a choice of number of factors.

The author is a visiting scientist from the *Institut National des Telecommunications* - 91000 EVRY - FRANCE.

Let us recall that a fractional factorial design is said to be :

- *saturated* if the number of observations (runs) is equal to one plus the number of factors.
- *orthogonal* if the covariance between any two estimable effects is zero.
- *of resolution 2t+1* if under the usual model all effects of order t or less are estimable whenever all effects of order higher than t are assumed to be zero.

To construct Hadamard matrices, we shall employ seven different methods. We first briefly describe each of these methods via an example and give the main function call for its corresponding implementation. We then give the general algorithm we use to generate those specific matrices. Discussion ends with a section devoted to the way these matrices need to be transformed before they can be used in the "Synthetic Perturbation Tuning" technique.

2. Methods

2.1 Notations and definition

From now on we adopt the following notation :

i) I denotes the identity matrix.
ii) $\mathbf{1}$ denotes a column vector with all entries +1
iii) A' and I' denote transposes of matrices A and I.

A square matrix A of order n whose entries are **+1** or **-1** is called a ***Hadamard matrix of order n*** provided that its rows are pairwise orthogonal, in other words $AA'=nI$. Note that throughout the report **'+'** and **'-'** will be used as abbreviation for **'+1'** and **'-1'**.

A matrix A of order n is said to be :

- *circulant* if its $A_{i+1,j+1}$ entries are equal to its $A_{i,j}$ entries (row and column numbers are reduced modulo n when i and/or j equals n). Hence the whole matrix is determined by its first row.
- *forward circulant* if its $A_{i+1,j+1}$ entries are equal to its $A_{i,j}$ entries (or $A_{i,j-i+1} = A_{i,j}$).
- *backward circulant* if its $A_{i+1,j-1}$ entries are equal to its $A_{i,j}$ entries (or $A_{i,i+j-1} = A_{i,j}$).

First rows to generate circulant matrices are stored in static arrays. When needed, that information is loaded in a dynamic array *Generator*. Throughout the programs, *Generator* will be the reference structure used to generate circulant matrices.

2.2 Method A

This first method may be used to construct Hadamard matrices of order 2^k for every positive integer k.

Let $\otimes$ denote the direct product of matrices. If A is the matrix with typical entry a_{ij}, then :

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots \\ a_{21}B & a_{22}B & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix}$$

If A is a Hadamard matrix of order m and B is a Hadamard matrix of order n then $A \otimes B$ is a Hadamard matrix of order mn. Hadamard matrices of order 2^k for every positive integer k, can be constructed by repeated use of the Hadamard matrix of order 2 as B. The Hadamard matrix of order 2 we shall use thereafter is

$$B = \begin{bmatrix} + & + \\ + & - \end{bmatrix}$$

Example : Construction of a Hadamard matrix of order 8.

Step 1 : Allocate space S for the Hadamard matrix.

S =

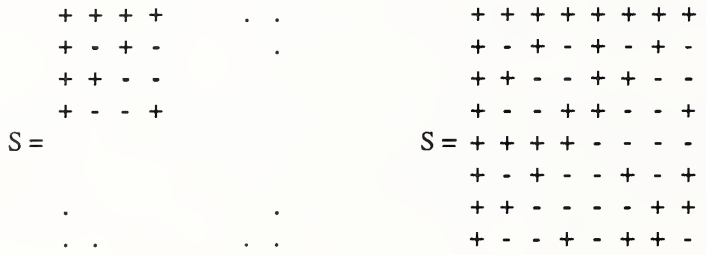
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .

Step 2 : Initialize the upper left block of the Hadamard matrix to the elements of the Hadamard matrix of order 2. This particular block is refered to as $\mathbf{S_a}$.

S =

+ +
+ -

Step 3 : Iteratively take the direct product $S = S_a \otimes B$ until the desired order is reached. Note that the order of S_a doubles after each iteration.



1st iteration 2nd iteration

Implementation : Hadamard matrices whose order ***Order*** is a power of 2 may be constructed by a call to ***Hadamard_corol31(Order)***. The direct product by the Hadamard matrix of Order 2 is performed by ***Direct_Product_by_Order2(element,matrix,i,j)***, ***element*** being the element to be multiplied and ***(i,j)*** being its coordinates in the matrix ***matrix***.

2.3 Method B

Now, suppose p is a prime number and $(p+1) \equiv 0 (mod\ 4)$, then there is a Hadamard matrix of order $(p+1)$. *(Unlike [HW77], we will be dealing with prime numbers p instead of prime powers p^{α} as we are working with a category of constructive methods labeled arithmetical [ACM64], p.393).*

Let Q be a forward circulant matrix of order p. The (i,j) entry of Q equals $\chi(j-i)$, *where* χ is a quadratic character namely :

$$\begin{cases} \chi(0) = 0 \\ \chi(b) = 1 \text{ if } b \textit{ is a nonzero quadratic residue} \\ \chi(b) = -1 \text{ if } b \textit{ is not quadratic.} \end{cases}$$

Let p be a prime number. If $x^2 = b (mod\ p)$, for positive integer x, then b is called a ***quadratic residue of p***. Among the elements $1,\ldots,p-1$ there are exactly $(p-1)/2$ quadratic residues of p.

Writing

$$S = \begin{bmatrix} 0 & 1' \\ -1 & Q \end{bmatrix}$$

H=I+S is a Hadamard matrix

Example : As 12=11+1 where $p = 11$ is a prime and $(p + 1) = 12 \equiv 0(mod\ 4)$ the current method can be used to construct a Hadamard matrix of order 12.

Step 1 : Allocate space for the Hadamard matrix

Step 2 : Find the non-zero quadratic elements (perfect squares) in the field F(Order-2)=0..(Order-2), here F(10).

Non-zero quadratic elements are stored in a one-dimensional array *Generator*, their corresponding field being set to 1.

In our example, the non-zero quadratic elements being 1, 3, 4, 5, 9 *Generator* is initialized as follows

Fields	*g0*	*g1*	*g2*	*g3*	*g4*	*g5*	*g6*	*g7*	*g8*	*g9*	*g10*
Generator	-	+	-	+	+	+	-	-	-	+	-

$0^2 = 0$ $4^2 = 16 \equiv 5(mod\ 11)$ $8^2 = 64 \equiv 9(mod\ 11)$
$1^2 = 1$ $5^2 = 25 \equiv 3(mod\ 11)$ $9^2 = 81 \equiv 4(mod\ 11)$
$2^2 = 4$ $6^2 = 36 \equiv 3(mod\ 11)$ $10^2 = 100 \equiv 1(mod\ 11)$
$3^2 = 9$ $7^2 = 49 \equiv 5(mod\ 11)$

Step 3 : Fill the first row with +1 and the first column except the first element with -1.

```
+ + + + + + + + + + + +
-
-
-
-
-
-
-
-
-
-
-
```

Step 4 : Q is a *forward circulant* matrix the first row of which is *Generator*, hence it is obtained by developing its first row modulo n (i.e 12). The diagonal is then set to 1 (This holds because H=I+S).

```
+ + + + + + + + + + + +
- + + - + + + - - - + -
- - + + - + + + - - - +
- + - + + - + + + - - -
- - + - + + - + + + - -
- - - + - + + - + + + -
- - - - + - + + - + + +
- + - - - + - + + - + +
- + + - - - + - + + - -
- + + + - - - + - + + -
- - + + + - - - + - + +
- + - + + + - - - + - +
```

Implementation : Those matrices may be constructed by a call to **Hadamard_th32(Order)**,*Order* being the order of the matrix to be built. The *Generator* is initialized by a call to **Create_Generator(Order)**.

2.4 Method C

We now consider Hadamard matrices of order $2(p + 1)$. For such a matrices to exist the following condition must hold : p is a prime number and $(p + 1) \equiv 2 (mod\ 4)$.

This method of construction is very similar to the previous one. Using the quadratic character χ, a forward circulant matrix Q is built as in method B. If

$$S = \begin{bmatrix} 0 & 1 \\ 1' & Q \end{bmatrix}$$

then

$$H = S \otimes \begin{bmatrix} + & + \\ + & - \end{bmatrix} + I \otimes \begin{bmatrix} + & - \\ - & - \end{bmatrix}$$

is the required Hadamard matrix.

Example : As $p = 5$ is a prime number and 5+1=2(mod 4) this method can also be used to construct a Hadamard matrix of order 2(5+1)=12.

Step 1 : Allocate space for the Hadamard matrix.

Step 2 : Find Q's first row i.e, initialize *Generator*.

In the field 0..4, non-zero quadratic elements are 1 and 4 therefore *Generator* is :

Fields	*g0*	*g1*	*g2*	*g3*	*g4*
Generator	-	+	-	-	+

Step 3 : Work on the first and second row. Fields with an even column number in the first row are set to +1. Remaining gaps are filled by processing the direct product by the Hadamard order 2 matrix on the previous elements (all the +'s in the first two rows).

```
+ . + . + . + . + . + .          + + + + + + + + + + + +
                                 + - + - + - + - + - + -
```

```
.                   .
. .                 . .          .                   .
                                 . .                 . .
```

Step 4 : Proceed as in step 3 on the two first columns.

```
+ + + + + + + + + + + +
+ - + - + - + - + - + -
+ +
+ -
+ +
+ -
+ +
+ -
+ +
+ -                                                   .
+ +                   .
+ -                 . .
```

Step 5 : Work now on two columns at a time. Columns identified with an even number are the working ones. Fields with an even row number are set to their corresponding value of the generator array (this yield to construct the forward circulant matrix Q). Columns identified with an odd number and the remaining fields are filled by application of the direct product on each of the previous values.

Working on columns 2 and 3, this yields :

```
+ + + + + + + + + + + + + +
+ - + - + - + - + - + - + -
+ + -
+ - -
+ + +
+ - -
+ + -
+ - -
+ + -
+ - -
+ + +                   .
+ - -                 . . .
```

```
+ + + + + + + + + + + + + +
+ - + - + - + - + - + - + -
+ + - -
+ - - +
+ + + +
+ - + -
+ + - -
+ - - +
+ + - -
+ - - +
+ + + +                   .
+ - + -                     .
```

Once the work is completed :

```
+ + + + + + + + + + + + + +
+ - + - + - + - + - + - + -
+ + - - + + - - - - + + - -
+ - - + + - - + - + + - + -
+ + + + - - + + - - + + - -
+ - + - - + + - - + + - + -
+ + - - + + - - + + - + - +
+ - - + + - - + + - - - - -
+ + - - - - + + - - - - -
+ - - + - + + - - + - + - +
+ + + + - - - - + + - - - -
+ - + - - + - + + - - + - -
```

Step 6 : Fill the diagonal with

```
+ -
- -
```

This holds because

$$H = H1 + I * \begin{bmatrix} + & - \\ - & - \end{bmatrix}$$

```
+ - + + + + + + + + + +
- - + - + - + - + - + -
+ + + - + + - - - - + +
+ - - - + - - + - + + -
+ + + + + - + + - - + +
+ - + - - - + - - + + -
+ + - - + + + - + + - +
+ - - + + - - - + - - -
+ + - - - - + + + - - -
+ - - + - + + - - - - +
+ + + + - - - - + + + -
+ - + - - + - + + - - -
```

Implementation : The corresponding call is ***Hadamard_th33(Prime_Number)***, where *Prime_Number* is a prime that is 5 according to our example.

2.5 Method D

Hadamard matrices for order **4n**, odd $n \leq 29$ and $n = 43$ may be obtained by using the Williamson's type method.

Consider the array :

$$H = \begin{bmatrix} A & B & C & D \\ -B & A & -D & C \\ -C & D & A & -B \\ -D & -C & B & A \end{bmatrix}$$

A, B ,C and D may be chosen in such a way that H will be a Hadamard matrix. A, B,C and D are *forward circulant* matrices *ie* they are entirely defined by their first row.

Example : Construct once again a Hadamard matrix of order 12.

Step 1 : Allocate space for the Hadamard matrix.

Step 2 : Fill each of the 16 blocks with the corresponding circulant matrix. First rows for those matrices are respectively :

A : + - -
B : + - -
C : + - -
D : + + +

Therefore the Hadamard matrix is

```
+ - - + - - + - - + + +
- + - - + - - + - + + +
- - + - - + - - + + + +
- + + + - - - - - + - -
+ - + - + - - - - - + -
+ + - - - + - - - - - +
- + + + + + + - - - + +
+ - + + + + - + - + - +
+ + - + + + - - + + + -
- - - - + + + - - + - -
- - - + - + - + - - + -
- - - + + - - - + - - +
```

Implementation : The corresponding call is ***Hadamard_William(Order)*** *Order* being the prime number p.

First rows for forward circulant matrices A, B, C and D which may be used to construct Hadamard matrices of order $4n$, for ***n=3, 5, 7, 13, 19, 23, 25, 29 and 43*** are listed in the table below:

	n=3	n=5	n=7	n=13
A:	+--	++--+	++-+++-	++---+--+---+
B:	+--	+-++-	++----+	++---+--+---+
C:	+--	+----	++-+++-	+---+-++-+---
D:	+++	+----	+-+++++-	++++-+--+-+++

n=19

```
A:  +-+---++++++++---+-
B:  ++-++--+-+++-+--++-+
C:  ++--+++-+--+-+++--+
D:  ++--+-++++++++++--+
```

n=23

```
A:  ++---+---+-+++-+---+---+
B:  +-+++-+++--++++++--+++-++-
C:  +++---++-+-+++-+-++---++
D:  +++-+++-+------+-+++-++
```

n=25

```
A:  +--+++-+-+--------+-+-+++--
B:  +-+++-+++-++----++-+++-++-
C:  +++-+++---+-++-+---+++-++
D:  +-++++----++-- ++----++++-
```

n=29

```
A:  ++--+--+-+++-++++-+++-+--+--+
B:  ++++-++-+---++++++---+-++-+++
C:  +++---++--+-+----+-+--++---++
D:  +-+---++--+-++++++-+--++---+-
```

n=43

```
A:  +---++--++++-+-+++-++--++-+++-+-++++--++---
B:  ++-++++++----+-+--++-++-++--+-+----++++++-+
C:  +++-+-++---+-+-++++-+----+-++++-+-+--++-+-++
D:  ++---++++-+--+--++--------++--+--+-++++---+
```

Contribution made by Ching-Shui Cheng of the University of California, Berkeley in identifying the forward circulant matrices for n=43 from Williamson's paper

2.6 Method E

This method is specifically used to build a Hadamard matrix of order 156. To do so we use the following Baumert-Hall array of size 12.

$$
\begin{array}{cccccccccccc}
A & A & A & B & -B & C & -C & -D & B & C & -D & -D \\
A & -A & B & -A & -B & -D & D & -C & -B & -D & -C & -C \\
A & -B & -A & A & -D & D & -B & B & -C & -D & C & -C \\
B & A & -A & -A & D & D & D & C & C & -B & -B & -C \\
B & -D & D & D & A & A & A & C & -C & B & -C & B \\
B & C & -D & D & A & -A & C & -A & -D & C & B & -B \\
D & -C & B & -B & A & -C & -A & A & B & C & D & -D \\
-C & -D & -C & -D & C & A & -A & -A & -D & B & -B & -B \\
D & -C & -B & -B & -B & C & C & -D & A & A & A & D \\
-D & -B & C & C & C & B & B & -D & A & -A & D & -A \\
C & -B & -C & C & D & -B & -D & -B & A & -D & -A & A \\
-C & -D & -D & C & -C & -B & B & B & D & A & -A & -A
\end{array}
$$

A, B, C and D are *symmetric forward circulant* matrices, the first rows of which are respectively

A : + + - - + - + + - + - - +
B : + - - - + + + + + + - - -
C : + + + - + + - - + + - + +
D : + + - + - + + + + - + - +

Implementation : The implementation is very similar to the previous one. Each of the 12*12 blocks is replaced by its corresponding circulant matrix inside a loop. The function call is currently ***Hadamard_Baumert(Order)*** with ***Order=13***.

2.7 Method F

We will be using this method to construct the Hadamard matrix of order 188. Considering the following array :

$$\begin{bmatrix} A & B & C & D \\ -B & A & -E & F \\ -C & E & A & G \\ -D & -F & -G & A \end{bmatrix}$$

The Hadamard matrix is constructed by replacing A, B, C, D, E, F and G by their corresponding forward circulant matrices

First rows for those matrices are respectively :

```
A: +-+----+--+----++---++++---+-+----++++++--+---+
B: +---+--++++++----+-+---+---++++--++++-++-++++-+-
C: +--+--+-+++-----+---+---++--+-++-+++-+-+--+---+
D: +--+--+-+++-----+---+-----++-+--+---+-+-++-+++-
E: +++-++-+-+---+--+-++-----+---+-----+++-+--+--+-
F: ---+--+-+-+++-++-+--++---+---+-----+++-+--+--++
G: -+----+--+----++---+++-+++-+-+++++------++-+++-+
```

Implementation : One can construct this Hadamard matrix of order 188 by calling ***Hadamard_Turyn(Order)*** with ***Order=188***.

2.8 Method G

It is known that the direct product of two Hadamard matrices is a Hadamard matrix. This characteristic is here being used to build the matrices we have not been able to construct directly with either of the previous methods.

Example : A matrix of order 40 may be built by processing the direct product of a matrix of order 2 (Method A) with a matrix of order 20 (Method B).

Implementation : The function call for this last method is ***Order_Product(Order)***, ***Order*** being the order of the matrix to be built.

3. General algorithm

The present algorithm allows us to build Hadamard matrices of order up to 200. This is not in any way a theoretical limit but a practical one. The programs can easily be extended to generate matrices of higher orders but one has to keep in mind that it is not known whether Hadamard matrices exist for every n which is a multiple of 4. However, we are sure they do exist for orders up to 200.

Our algorithm is based on *Table 4* from [HW77], which we reproduce below. It uses a first-found first-used basis that is, according to the order of the matrix to be built, the first method whose premises are satisfied is used to build the matrix. No other method will thereafter be sought. However, different methods may be used to generate a Hadamard matrix of a given order.

Methods of construction used :

Methods of construction of Hadamard matrices orders up to 200			
1-A	48-B	100-D	152-B
2-A	52-D	104-B	156-E
4-A	56-AC	108-B	160-AB
8-A	60-B	112-AC	164-B
12-B	64-A	116-D	168-B
16-A	68-B	120-AB	172-D
20-B	72-B	124-C	176-AB
24-B	76-C	128-A	180-B
28-C	80-B	132-B	184-AD
32-A	84-B	136-AB	188-F
36-C	88-AB	140-B	192-B
40-AB	92-D	144-AB	196-C
44-B	96-AB	148-C	200-B

A : Constructed in Corollary 3.1

B : Constructed in Theorem 3.2

C : Constructed in Theorem 3.3

D : Constructed by Williamson's method

E : Constructed by Baumert and Hall

F : Constructed by Turyn using Baumert-Hall arrays

AB,AD : A product of orders, one constructable by method A and the other by method B; similarly for AD. (For programming purpose this method has been identified by G)

General Algorithm

= * =

Order being the order of the Hadamard matrix to be built :

{

If ***Order=1*** or ***Order=2*** use method A
If ***Order=156*** use method E
If ***Order=188*** use method F
If ***Order is a power of two use*** method A
If ***(Order-1) is a prime number and Order≡0mod(4)*** use method B
If ***(Order/2-1) is a prime number and Order/2≡2mod(4)*** use method C
If ***(Order/4) is an odd number*** use method D
Other cases : use method G

}

Implementation : To create any Hadamard matrix of order up to 200 invoke ***Create_Hadamard(Order)***, ***Order*** being the order of the desired matrix.

4. Adapting Hadamard Matrices for the Synthetic Perturbation Tuning Technique

Hadamard matrices may be viewed as plans of multifactorial experiments. Therefore, they may be used to generate the patterns of delays needed to test a program ***P*** in the "Synthetic Perturbation Tuning" technique of program sensitivity analysis.

Once locations of interest have been chosen in the program ***P***, patterns of delay treatments can be determined. In other words, given a certain number of delays (locations or factors) one can generate the needed Hadamard matrix. Factors correspond to matrix columns. Entries in the columns correspond to the test levels of the factors and rows correspond to the test runs. Each treated version of ***P*** is then run and its overall response ***R*** measured. By a comparison among significant effects one can next establish which factors most influence the response ***R***.

Response ***R*** varies about an overall mean μ. So as to be able to compute this mean, we require our matrices to have an entire column of 1. This may easily be obtained by seminormalizing the matrices. Note that, as a result, no factor is assigned to this particular column. Furthermore, in a matrix of a given order ***n*** at most ***n-1*** columns may be assigned a factor. Assuming now that the number of factors ***m*** is such that $m \leq n-1$, only the first ***m*** columns will be considered in the analysis (first column excluded).

In short, ***d*** being the number of delays of interest (locations or factors) in ***P***, the order ***n*** of the Hadamard matrix to be created is chosen as follows :

$$n \text{ is the first integer divisible by 4 such that } n \geq (d+1)$$

(***n*** has to be a multiple of 4 to be a possible order for a Hadamard matrix. In a matrix of order ***n*** at most ***(n-1)*** may be assigned a factor so, as a result $n \geq (d+1)$).

Example : Assuming 17 locations have been selected in our programs (*ie* 17 delays are to be inserted), the order of the Hadamard matrix to be used will be 20.

By construction, Hadamard matrices generated with Method A are normalized thus seminormalized, hence they need no modification.

Now, those matrices that do not match our requirement, can be seminormalized in a very straightforward manner, that is, by negating all the rows whose first element is a '-1' (i.e -1 becomes +1 and +1 becomes -1). This applies for all methods. However, with Method C there is no need to check the entire first column for '0' elements as only the second row needs to be negated. Furthermore, matrices generated with this particular method may easily be normalized by negating their second column.

In the particular case of Method G, we have two possible approaches. One is to seminormalize the matrices before performing the direct product, in which case, the resulting matrix would have been seminormalized. The second is to seminormalize the resulting matrix. For programming consistency we have chosen the second option. Whatever method has to be used, we first produce the Hadamard matrix and then seminormalize it.

Implementation : Any Hadamard matrix may be seminormalized by a call to *Norme_matrix(Hadamard,Order)*, *Hadamard* being the matrix of order *Order* to seminormalize. Hadamard matrices built with method C may also be seminormalized by a call to *Norme_th33(Hadamard,Order)*.

Example : Hadamard matrix of order 12 built with method D

```
+ - - + - - + - - + + +
- + - - + - - + - + + +
- - + - - + - - + + + +
- + + + - - + + + - + +
+ - + - + - + + + + - +
+ + - - - + + + + + + -
- + + - + + + - - + - -
+ - + + - + - + - - + -
+ + - + + - + + - - - +
- - - + - - - + + + - -
- - - - + - + - + - + -
- - - - - + + + - - - +
```

Matrix created with Method D

```
+ - - + - - + - - + + +
+ - + + - + + - + - - -
+ + - + + - + + - - - -
+ - - - + + - - - + - -
+ - + - + - + + + + - +
+ + - - - + + + + + + -
+ - - + - - - + + - + +
+ - + + - + - + - - + -
+ + - + + - + + - - - +
+ + + - + + + - - - + +
+ + + + - + - + - + - +
+ + + + + - - - + + + -
```

Seminormalized matrix

5. Acknowledgments

I would like to thank Dr. Raghu N. Kacker for giving the original inspiration and helping me with the theory on Hadamard matrices and Design of Experiments. I am especially grateful to Dr. Gordon Lyon for making this work possible and suggesting numerous details for improvements by reading earlier versions of the text. My thanks also go to Robert Snelick for his help throughout the overall work.

6. References

[LSK92] : G.Lyon, R.Snelick, R.Kacker, "Time-Perturbation Tuning of MIMD Programs," Performance Tools '92, R.Pooley and J.Hillston (eds.) Edinburgh University Press Ltd. Proceedings of the Sixth International Conference on Modeling Techniques and Tools for Computer Performance Evaluation , 16-18 September, 1992.

[KLF91] : R.N.Kacker, E.S.Lagergren, J.J.Filliben, "Taguchi's Orthogonal Arrays Are Classical Designs of Experiments," Journal of Research of the National Institute of Standards and Technology, Vol.96, No.5, Sept-Oct 1991.

[WW88] : W.D.Wallis, "Combinatorial Designs," Pure and Applied Mathematics, Marcel Dekker, INC 1988.

[B85] : T.B.Barker, "Quality by Experimental Design," Marcel DEKKER Inc, 1985.

[BHH78] : G.E.P Box, W.G.Hunter, J.S.Hunter, "Statistics for Experimenters," John Wiley & Sons, New York, 1978.

[HW78] : A.Hedayat, D.W.Wallis, "Hadamard matrices and their application," University of Illinois at Chicago Circle and University of Newcastle, The Annals of Statistics 1978, Vol.6, No.6, 1184-1238.

[B66] : L.D.Baumert, "Hadamard Matrices of order 116 and 232," Bull. Amer. Soc. 71, 169-170

[BH65a] : L.D.Baumert, M.Hall, Jr., "A new construction for Hadamard matrices," Bull. Amer. Math. Soc. (1965a) 71 169-171.

[BH65b] : L.D.Baumert, M.Hall, Jr., "Hadamard Matrices of the Williamson Type," Math. Comp. 19 442-447.

[ACM64] : "Applied Combinatorial mathematics," University of California engineering and physical sciences extension series, John Wiley & Sons, Inc 1964.

[BGH62] : L.D.Baumert, S.W.Golomb, M.Hall, Jr., "Discovery of an Hadamard Matrix of Order 92," Bull. Ameri. Math. Soc. 68 (1962), 237-238.