Paper 262-25

Custom Design of Complicated Block Experiments

G. Bruce Schaalje, Department of Statistics, Brigham Young University, Provo, UT
Chris H. Bodily, Department of Statistics, North Carolina State University, Raleigh, NC
Bruce Jay Collings, Department of Statistics, Brigham Young University, Provo, UT

ABSTRACT

There are many experimental design situations for which the available material does not meet the required treatment or blocking structure of a classical design. We present a SAS MACRO, based on PROC MIXED, which finds useful designs for virtually all situations involving experimental material classified by a single blocking variable. The procedure randomly constructs and examines a large number of possible designs based on the specified treatment and blocking structures. The design with the best value of an optimality measure is selected as an effective, if not fully optimal, design for the experiment. An example illustrates the use of the procedure.

INTRODUCTION

Many clever experimental designs are available for nice situations involving block-structured experimental material. These include classical balanced incomplete blocks, confounded designs for 2^n and 3^n factorial treatment structures, lattices, and others (Kuehl 2000). Though useful, these designs require such things as blocks of equal size, blocks of specified sizes depending on the number of treatments, specified numbers of blocks, equally-correlated units within each block, and non-mixed factorial treatment structures. More flexible modern designs drop the requirement for certain numbers of blocks of certain sizes (John and Williams 1995) or the requirement for equally-correlated units within each block (Morgan and Chakravarti 1988), but still typically require blocks of equal sizes and unstructured treatments.

In many situations, it is not possible for experimental material to meet the requirements of a classical design. As an example, in a recent experiment to compare 4 treatments for warts, the available experimental material consisted of warts on 5 cattle. Two had 3 warts each, and the other 3 cattle had 2, 4 and 6 warts, respectively. Warts were the experimental units for the study, and cows were the blocks. Obviously, it was not possible to arrange for a certain number of cattle to have a certain constant number of warts each.

"Designing an experiment" has sometimes been interpreted as picking from a library of designs the design recipe which comes closest to fitting the particular circumstances of the experiment, and then making compromises in the objectives of the experiment and the structure of the experimental material in order to force the experiment into the recipe's requirements. Classical designs are important, and it is surprising how often their requirements are or can be naturally met in practical situations. Nonetheless, many statisticians have recognized the need for approaching experimental design in the opposite way: constructing effective custom designs to fit the objectives and available experimental material (Mead 1989).

Although the use of computers in design of experiments was surprisingly a relatively late development (Mead 1989), many programs useful in designing experiments are now available either as stand-alone programs or as part of a statistics package. For example, the SAS® System provides the FACTEX and OPTEX procedures as well as the ADX interface. FACTEX and the ADX system greatly facilitate the selection and use of classical blocked and unblocked designs for many kinds of treatment structures, including mixed factorials. OPTEX searches for optimal designs for certain non-standard situations including treatment structures which exclude certain factor combinations, nonlinear models, and limitations on experimental materials. However, OPTEX is not user-friendly and does not seem to provide help in situations involving unequally-sized blocks or serially correlated units within blocks. Very general algorithms are available for theoretically finding optimal designs in virtually all experimental situations (Nguyen and Williams 1993, Miller and Nguyen 1994, John and Whitaker 1993), but these have yet to be made available in user-friendly and general form. It appears that these algorithms must be extensively modified for use in different design situations.

The purpose of this paper is to suggest a simple and yet very general approach to the design of complicated block experiments. A SAS MACRO for the approach is available from the authors.

BRUTE FORCE DESIGN OF EXPERIMENTS

The idea of our procedure is simple. A computer program randomly constructs a large number of possible designs, using the specified treatment structure and the available experimental units with their blocking structure. Dummy data are generated for each design, and the designs are analyzed using a mixed linear model. An optimality criterion is computed for each design. The design with the best value of the optimality measure is selected as an effective, if not fully optimal, design for the experiment. This procedure is admittedly not elegant. It simply relies on the brute force of the computer to generate a good design.

This procedure is feasible because powerful and fast computers with accompanying software are readily available. Hundreds of designs can be generated and analyzed in a short enough period of time to be practical. Furthermore, it turns out

that only a few hundred designs need to be generated; if as few as 500 designs are randomly produced, the probability exceeds 99% that a design in the top 1% of all designs will be generated.

Fully optimal designs will rarely be produced by this procedure, but efficient and nearly optimal designs will always be generated. Franklin and Payne (1993) call such designs "effective designs", claiming that they are as useful as fully optimal designs because experimenters rarely have sufficient knowledge to be able to exactly specify either the model or the appropriate optimality criterion. And even if they did, failures in the conduct of the design such as those resulting in missing values, would lead to loss of absolute optimality of the design. Furthermore, there may not be a single optimal design in complicated situations involving, for example, unequally sized blocks.

Brute force selection of an effective design is paradoxically both simpler and more general than other approaches. This is because the program can be built around a familiar but versatile data analysis procedure such as the MIXED Procedure. Data from virtually all designs (including unbalanced designs with unequally sized blocks, complicated treatment structures, and serially correlated units within blocks) can be easily and properly analyzed using PROC MIXED. Therefore, an algorithm that accepts initial data (such as the number and size of the blocks, type of covariance structure, etc.) can use PROC MIXED to generate information useful in the evaluation of a wide variety of potential designs. The method utilizes common statistical terminology, and familiar software. The generality of PROC MIXED transfers to the design algorithm.

This approach takes the focus of design of experiments away from non-statistical ideas (modulo arithmetic, Galois field theory, exchange algorithms, simulated annealing) that can dominate the subject, and returns it to the basic statistical idea of experimental design: given a statistical model, what sort of data would it generate, and with what properties (Nelder 1996)?

STATISTICAL DETAILS

The experimental designs of interest in this paper involve one blocking factor (a set of blocks of specified sizes), a set of treatment factors, and the proposed model for main effects and interactions of the treatment factors. The statistical model is $\mathbf{y} = \mathbf{Xb} + \mathbf{Zg} + \mathbf{e}$ where $\mathbf{y}$ is a vector of responses, $\boldsymbol{\beta}$ is a vector of unknown parameters for the treatment effects, $\mathbf{g}$ is a vector of unknown random block effects, $\mathbf{e}$ is a vector of unknown random errors, $\mathbf{X}$ is a design matrix connecting the responses to the treatment effects, and $\mathbf{Z}$ is a design matrix connecting the responses to the block effects. While $\mathbf{g}$ and $\mathbf{e}$ represent unknown effects, it is assumed that $\mathbf{g} \sim N(\mathbf{0}, \sigma_b^2\mathbf{I})$ and $\mathbf{e} \sim N(\mathbf{0}, \mathbf{R})$. If the units within each block are equally correlated, $\mathbf{R} = \sigma_\varepsilon^2\mathbf{I}$; otherwise $\mathbf{R}$ is a block diagonal matrix determined by the sizes and correlation structure of the experimental blocks. In either case, σ_b^2 and parameters of $\mathbf{R}$ are assumed to be known and must be supplied by the researcher. $\mathbf{X}$ and $\mathbf{Z}$ are dependent on the assignment of treatments to experimental units within the blocks, and therefore change among randomly generated designs.

The variance matrix of $\mathbf{y}$ is the block-diagonal matrix $\mathbf{V} = \sigma_b^2\mathbf{ZZ}' + \mathbf{R}$. Given $\mathbf{V}$ for a particular design, $\mathbf{b}$ is estimated as $\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}'\mathbf{V}^{-1}\mathbf{y}$ with variance matrix $\Sigma = (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-1} = [\mathbf{X}'(\sigma_b^2\mathbf{ZZ}'+\mathbf{R})^{-1}\mathbf{X}]^{-1}$. The objective of our design algorithm is to find the allocation of units to treatments such that some relevant aspect of $\mathbf{S}$ is optimized (at least among the randomly generated designs).

A popular method of selecting an optimal design chooses that design which minimizes $q = \det[\mathbf{S}]$. Designs declared as optimal by this method are referred to as *D*-optimal designs. Other criteria for optimizing designs have also been suggested, such as *A*-optimality and (*M,S*)-optimality based on other functions of $\mathbf{S}$ (John and Whitaker, 1993). Our program currently uses *D*-optimality, but can be modified to use other criteria.

IMPLEMENTATION ON THE SAS SYSTEM

The SAS MACRO (currently written in SAS 6.12) used to find effective designs is called BRUTE. Full details on the MACRO are given by Bodily (1996). Input to BRUTE includes the number of designs to be generated, the number of treatment factors, the number of levels of each factor, the model for the treatment effects (main effects and interactions), the maximum number of times a given treatment can appear in the design, number of blocks, size of each block, and the within-block covariance structure along with the necessary parameter values (such as σ_b^2 and σ_ε^2). BRUTE uses this information to create a command file which will generate and analyze designs matching the specifications.

BRUTE randomly assigns treatments directly to the experimental units, ignoring block boundaries, and restricts the maximum number of times a given treatment can be assigned. Once a treatment is distributed the maximum number of times, it is removed from the pool of usable treatments. By explicitly requiring that treatments appear no more than some maximum, BRUTE implicitly requires that each treatment also appears some minimum number of times. Control of the maximum is given to the user, but BRUTE requires the maximum multiplied by the total number of treatments to be greater than or equal to the total number of experimental units and will automatically increase the maximum if necessary. The minimum number of times a treatment appears is controlled by careful selection of the maximum. Setting the maximum too high decreases the performance of BRUTE by increasing the total number of design configurations possible.

Once a random design is generated, it is saved as a SAS data set. PROC MIXED then reads in the treatment and block assignments and constructs $\mathbf{X}$, $\mathbf{Z}$ and something related to $\mathbf{S}$. Since PROC MIXED was intended for data analysis, some interesting challenges arise when using it for design. First, one has to supply PROC MIXED with response data, so a vector of random data with the proper number of observations is generated. Also, BRUTE disables the variance matrix parameter estimation algorithm and requires PROC MIXED to use values given by the user. Implementing user-specified parameter values, however, is not as straightforward as the PROC MIXED documentation suggests. The *noiter* option used in conjunction with the *sigiter* option and a *repeated* statement make PROC MIXED recognize the user-provided variance parameters. Another problem arises when only one block is used in a design, as in a completely random or fractional factorial design, and the blocking variable contains only one subscript. A solution to this problem requires deleting both the *subject* component of the *repeated* statement and the entire *random* statement. PROC MIXED no longer makes use of the among block covariance

parameter values, and they must be deleted from the *parms* statement.

The output delivery system of PROC MIXED allows one to save the 'variance matrix' of the parameters, but this is not exactly **S**. In its calculations, PROC MIXED creates a new matrix **W** = [**X**|**y**]. $\mathbf{W}'\mathbf{V}^{-1}\mathbf{W}$ is inverted using the sweep algorithm (Goodnight, 1979) and the result is the 'variance matrix' that can be saved as a SAS data set. Also, the **X** matrix generated by PROC MIXED corresponds to an overparametrized model (Searle 1987) and thus $(\mathbf{W}'\mathbf{V}^{-1}\mathbf{W})^{-}$ contains extra rows and columns of zeros, as well as a border row and column due to appending the data vector. This complicates design assessment using the *D*-optimality criterion.

BRUTE cleans up the saved matrix by removing the extra border column and row, and then traversing the matrix diagonal, deleting rows and columns corresponding to diagonal elements of zero. Even after cleaning the matrix, complications remain as the true variance matrix could have contained a "real" zero on the diagonal because one or more of the effects specified in the model statement was not estimable for the particular assignments of treatments to experimental units. This is impossible to distinguish from a zero due to overparametrization. Despite this problem, the determinant of the cleaned-up variance matrix is calculated. BRUTE then makes a comparison between the model degrees of freedom for the full specified model and the rank of the current cleaned-up variance matrix. If they agree, all is well and θ is saved to be compared with previously calculated values of θ. If the degrees of freedom disagree, the current θ is set to zero. A non-zero θ is required before an overall design iteration is considered complete.

BRUTE continues until the specified number of design iterations have been completed. The current value of θ is compared to the best design already constructed. When a new design is found with a smaller value of θ than any previous design, it is stored to be used as the new standard of comparison. Thus, the program keeps a record of the best design found during a given session while continuing to search for better designs. After the design iterations are completed, the q values are summarized using the UNIVARIATE procedure, and the best design is printed.

BRUTE does not return results instantaneously, but is fast enough to conveniently run in the background of a computer. On a laptop (Pentium® II, 233 MHz, 128 MB), the program required about 6 minutes to run 500 iterations for a design with an 8 degree-of-freedom treatment structure and 36 experimental units in 12 equally sized blocks. The program required about 12 minutes for a design with a 21 degree-of-freedom treatment structure and 30 experimental units in 9 unequally sized blocks.

EXAMPLE

BRUTE has been applied to several design problems (Bodily 1996) including those for which classical designs are known to be optimal. Designs produced by BRUTE for these situations were as efficient or nearly as efficient as the classical designs. We give an example here to demonstrate the range of capabilities of BRUTE. The program was used to search for an effective design for a 2x3x5 factorial treatment structure applied to units in 9 blocks of unequal sizes varying from 2 to 5. Interest was in main effects and all 2-factor interactions of the factors. In this example, 5000 design iterations were used out of curiosity. Obviously, this design is so complicated that no comparison to a classical design is possible. Input to BRUTE included:

Treatment factors: 2 3 5, **Blocks**: 2 2 2 3 3 4 4 5 5, **Model**: A B C A*B A*C B*C
Covariance structure: Standard, $\sigma_b^2 = 5$, $\sigma_\varepsilon^2 = 1$,
Design iterations: 5000

Following is output from BRUTE, including the selected design and a summary of the q values. Note that the best design was generated as the 688th design iteration.

OBS	BLOCK	A	B	C	OBS	BLOCK	A	B	C
1	1	2	2	3	16	6	2	3	2
2	1	1	3	3	17	7	2	3	3
3	2	2	2	2	18	7	1	3	5
4	2	1	2	5	19	7	2	1	1
5	3	1	3	1	20	7	1	1	2
6	3	2	3	4	21	8	1	1	1
7	4	2	1	2	22	8	1	1	5
8	4	1	1	4	23	8	2	1	3
9	4	1	2	4	24	8	2	3	5
10	5	2	2	5	25	8	1	3	2
11	5	1	2	1	26	9	1	1	3
12	5	2	3	1	27	9	2	2	1
13	6	2	2	4	28	9	2	1	5
14	6	1	2	3	29	9	2	1	4
15	6	1	2	2	30	9	1	3	4

Extremes

Lowest	Obs	Highest	Obs
0.413831(	688)	102.8177(	42)
0.435116(	3365)	105.5080(	1470)
0.437262(	3404)	109.4355(	694)
0.438792(	1721)	111.6589(	678)
0.464012(	1031)	119.4270(	287)

```
                Histogram                    #  Boxplot
117.5+*                                      1     *
      .*                                     1     *
      .*                                     2     *
      .*                                     1     *
      .                                            
      .*                                     4     *
      .*                                     1     *
      .*                                     2     *
      .*                                     2     *
      .*                                     4     *
      .*                                     6     *
      .*                                     7     *
      .*                                     9     *
      .*                                     5     *
      .*                                    14     *
      .*                                    24     *
      .*                                    30     0
      .**                                   56     0
      .***                                  69     0
      .****                                105     0
      .********                            225     |
      .**********                          429   +---+
      .*********************************   989   *-+-*
  2.5+************************************************ 1418  +---+
     ----+----+----+----+----+----+----+----+----+----
```

* may represent up to 30 counts

It is not possible to know if the design produced by BRUTE is the optimal design for this situation, or even if a single optimal design exists. Nonetheless an examination of the within-block variances, for randomly generated data, of a series of contrasts can reveal if the design has some of the properties expected in an effective design. In an effective design, the variances of the contrasts should be approximately uniform. However, because they are calculated from randomly generated data, they may be different from effect to effect. As can be seen from the output below, the variances approximate this ideal. Due to the complex nature of the design specifications, it may not have been possible to come closer than this to the ideal.

effect	contrast	variance
A	1	1.77
B	1	1.94
	2	3.62
C	1	3.22
	2	2.63
	3	2.27
	4	4.51
A*B	1 1	1.00
		1.37
A*C	1 1	3.18
	1 2	2.87
	1 3	3.52
	1 4	2.70
B*C	1 1	8.84
	1 2	4.81
	1 3	2.39
	1 4	5.29
	2 1	6.67
	2 2	3.17
	2 3	3.38
	2 4	8.45

CONCLUSION

BRUTE is a useful tool in the design of certain complicated experiments. Work is currently underway to convert the MACRO to run on SAS 7.0. Other planned enhancements include the capability to deal with more than one blocking system, the inclusion of additional optimality criteria, and support for varying the levels of interest in specific contrasts.

REFERENCES

Bodily, C.H. (1996), *Brute Force Design of Confounded Experiments*, M.S. Thesis, Department of Statistics, Brigham Young University.

Franklin, M. F. And Payne, R. W. (1993), "Tools for the Construction of Effective Experimental Designs," *Proceedings of the 1993 Kansas State University Conference on Applied Statistics in Agriculture*.

Goodnight, J. H. (1979), "A Tutorial on the Sweep Operator," *American Statistician*, 33, 149-158.

John, J.A. and Whitaker, D. (1993), "Construction of Resolvable Row-Column Designs Using Simulated Annealing," *Australian Journal of Statistics*, 35(2), 237-245.

John, J.A. and Williams, E.R. (1995), *Cyclic and Computer Generated Designs*, London: Chapman and Hall.

Kuehl, R.O. (2000), *Design of Experiments: Statistical Principles of Research Design and Analysis*, 2nd ed., Belmont California: Duxbury Press.

Mead, R. (1988), *The Design of Experiments*, New York: Cambridge University Press.

Miller, A. J. and Nguyen, Nam-Ky (1994), "A Fedorov Exchange Algorithm for D-optimal Design," *Applied Statistics*, 43(4), 669-678.

Morgan, J.P and Chakravarti, I.M. (1988), "Block Designs for First and Second Order Neighbor Correlations," *Annals of Statistics*, 16, 1206-1224.

Nelder, J.A. (1996), "Generalized Linear Models," *Notes of the 1996 Summer Institute of Applied Statistics*, Brigham Young University.

Nguyen, Nam-Ky and Williams, E.R. (1993), "An Algorithm for Constructing Optimal Resolvable Row-Column Designs," *Australian Journal of Statistics*, 35(3), 363-370.

Searle, S. R. (1987), *Linear Models for Unbalanced Data*, New York: John Wiley and Sons.

CONTACT INFORMATION

G. Bruce Schaalje
Brigham Young University
223A TMCB
Provo Ut 84602
Phone (801) 378-3996
Fax (801) 378-5722
E-mail **schaalje@byu.edu**