Paper 3148-2015

Catering to Your Tastes: Using PROC OPTEX to Design Custom Experiments, with Applications in Food Science and Field Trials

Clifford Pereira, Department of Statistics, Oregon State University;
Randy Tobias, SAS Institute Inc.

ABSTRACT

The success of an experimental study almost always hinges on how you design it. Does it provide estimates for everything you're interested in? Does it take all the experimental constraints into account? Does it make efficient use of limited resources? The OPTEX procedure in SAS/QC® software enables you to focus on specifying your interests and constraints, and it takes responsibility for handling them efficiently. With PROC OPTEX, you skip the step of rifling through tables of standard designs to try to find the one that's right for you. You concentrate on the science and the analytics and let SAS® do the computing.

This paper reviews the features of PROC OPTEX and shows them in action using examples from field trials and food science experimentation. PROC OPTEX is a useful tool for all these situations, doing the designing and freeing the scientist to think about the food and the biology.

INTRODUCTION

Food scientists use a wide variety of designs, including factorial treatment designs for testing recipes and nested block designs for taste tests. Experimental blocks are designed to be as homogeneous as possible; they might consist, for example, of taste assessments of a set of samples by a particular judge in a particular session. The basic methods for block designs were pioneered in agricultural field trials. In food science as in field trials, it is often not possible to test all treatments in each natural homogeneous block. In a field trial with many treatments, a block of plots large enough to include all treatments would be too heterogenous to be useful. In food science, requiring each judge to assess all food samples in a single session would lead to fatigue and bias.

Both food scientists and agricultural researchers are also often concerned with *resolvability* in block designs A design is resolvable when multiple blocks can be combined in such a way that each treatment occurs equally often in each collection of blocks. If each treatment occurs once in each such collection of blocks, then each collection is said to be a replicate. Grouping into complete replicates is useful in managing an experiment and also in handling large-scale nuisance factors. In particular, resolvable designs are often used in field trials that include many treatments. In such trials, blocking is a key technique for accurately estimating treatment effects, and resolvability is in turn key for managing experiments that include many blocks on a replication-by-replication basis (Kuehl 2000). With a resolvable design, you can conduct the experiment in stages, with complete replicates at each stage, while still capturing local within-stage variation by using incomplete blocks. Resolvability can likewise be useful in multisite experiments and in experiments where multiple individuals handle experimental runs (Morgan and Reck 2007).

The purpose of this paper is to show how you can use the OPTEX procedure in SAS/QC software to easily find good resolvable designs that include any number of equally or unequally sized blocks within each replicate. Using this approach, you are freed from forcing your experiment into standard forms, enabling you to design experiments that include blocking structures customized for your situation. Given such structures, you can trust PROC OPTEX to turn out a very efficient resolvable design. The approach discussed can also be used in more general settings with nested blocks, as indicated in the final section.

ALL YOU REALLY NEED TO KNOW

Example 1: Wine Tasting

Suppose you are a food researcher who is studying wine preferences. You want your panel of 20 wine experts to conduct taste tests of 51 varieties of wine. Obviously, they cannot taste all 51 wines in the same session, at least not if you want to trust the ratings of the last dozen or so! So you propose to present the wines to them in three sessions of 17 wines apiece. You have two goals for the design:

1. Each expert should judge each wine once. (That is, each judge constitutes a replicate.)
2. Each pair of wines should occur in the same session in as balanced a way as possible.

The following DATA steps create two data sets that describe, respectively, the wine varieties that you are studying and the expert/session setup that you have decided on:

```
data Wines;
   do Wine = 1 to 51;
      output;
   end;
run;

data Setup;
   do Subject = 1 to 20;
      do Session = 1 to  3;
         do Plot = 1 to 17;
            output;
         end;
      end;
   end;
   drop Plot;
run;
```

The following statements invoke the OPTEX procedure to construct a good experimental design for this study:

```
proc optex data=Wines coding=orthcan seed=16899;
   class Wine;
   model Wine;
   block design=Setup;
   class Subject Session;
   model Subject, Session(Subject) / prior=0,10;
   output out=Design;
run;
```

The PROC OPTEX syntax describes the roles of the data sets in defining the “treatment” and “block” models: the first CLASS statement and MODEL statement define the treatment model, referring to the **Wine** variable in the input data set Wine, and the second CLASS and MODEL statements define the block model in terms of the **Subject** and **Session** variables in the Setup data set. The PRIOR= option in the second MODEL statement is the essential feature that enables the resulting arrangement to satisfy the two goals.

Finally, the %RDBEval macro, which is documented in “APPENDIX: %RBDEVAL MACRO” on page 10, evaluates the resulting design for efficiency with respect to replicates and blocks within replicates:

```
%RBDEval(Design,Wine,Subject,Session);
```

A D-efficiency rating of 100% with respect to replicates indicates that the design is resolvable. The D-efficiency rating with respect to blocks within replicates indicates how well-balanced the design is. For a discussion of the D-optimality criteria, see the section “Optimality Criteria” in the chapter “The OPTEX Procedure” of the *SAS/QC User’s Guide*.

The PROC OPTEX output in Figure 1 shows that the best design found has a Bayesian treatment D-efficiency of 96.63.

Figure 1 Example 1 PROC OPTEX Output: Bayesian Design Criteria (PRIOR=0,10)

The OPTEX Procedure

Design Number	Treatment D-Efficiency	Treatment A-Efficiency
1	96.6321	96.6085
2	96.6321	96.6085
3	96.6318	96.6080
4	96.6318	96.6080
5	96.6317	96.6078
6	96.6316	96.6075
7	96.6314	96.6072
8	96.6314	96.6071
9	96.6314	96.6071
10	96.6313	96.6070

This Bayesian efficiency crierion can be difficult to interpret directly, because it combines the D-efficiency with respect to replicates and the D-efficiency with respect to blocks within replicates in a particular way as defined by the PRIOR= specification. The output of the %RBDEval macro, shown in Figure 2, untangles these two efficiencies.

Figure 2 Example 1 Macro Output: Design Efficiency Evaluation for Each Structure

Wine Efficiency

Relative to Upper Bound for a Variance-Balanced Design

Evaluation	D-Efficiency
Subject	100.0000
Session(Subject)	99.9646

Figure 2 shows that the resulting design has 100% efficiency relative to subjects, so the design is resolvable and satisfies the first goal precisely. Figure 2 also shows that the design has a session-within-subjects efficiency of more than 99.9%, relative to an upper bound for a hypothetical balanced incomplete block design (BIBD) in which each pair of wines is evaluated together equally often. It is easy to show that such a balanced design cannot exist for this setting. However, PROC OPTEX finds a design that is close to being balanced and cannot be substantially improved upon because its efficiency is so close to the upper bound. Each pair of wines is tasted together between 3 and 10 times. Most pairs of wines are tasted together 6 or 7 times, and 90% of the pairs are tasted together between 5 and 8 times.

The remainder of this paper reviews the OPTEX procedure and explains why this method works.

RESOLVABLE DESIGNS

Example 2: A Small Resolvable Balanced Incomplete Block Design (BIBD)

Figure 3 shows a resolvable block design for four treatments in six blocks of size 2; the pairs of blocks in each row of the table make up a single replicate of the treatments. Because the blocks of this design consist of all pairs of treatments, it is also balanced and therefore is a resolvable balanced incomplete block design.

Figure 3 A Resolvable Balanced Incomplete Block Design

	Block	
Replicate	**1**	**2**
1	[1 2]	[3 4]
2	[1 3]	[2 4]
3	[1 4]	[2 3]

Resolvable balanced incomplete block designs are D-optimal with respect to two different types of blocks: the primary blocks and the replicates, considered as blocks. To confirm the optimality for the design in Figure 3, the following

statements use the %RBDEval macro to evaluate this arrangement of treatments, both as a design for six blocks of size 2 and as a design for three blocks of size 4.

The following DATA step creates a data set Design which contains a set of points for the design that is to be evaluated for D-efficiency:

```
data Design;
   do Replicate = 1 to 3;
      do Block = 1 to 2;
         do Plot = 1 to 2;
            input Treatment @@;
            output;
         end;
      end;
   end;
datalines;
1 2   3 4
1 3   2 4
1 4   2 3
;
```

The following statement calls the %RBDEval macro, which evaluates the design:

```
%RBDEval(Design,Treatment,Replicate,Block);
```

Figure 4 displays the results.

Figure 4 Example 2 Macro Output: Design Efficiency Evaluation for Each Structure

Treatment Efficiency
Relative to Upper Bound for a Variance-Balanced Design

Evaluation	D-Efficiency
Replicate	100.0000
Block(Replicate)	100.0000

The block design D-efficiency is 100% with respect to both block structures, confirming that the design is a resolvable balanced incomplete block design.

SEARCHING FOR A RESOLVABLE DESIGN

Example 3: Fifteen Treatments in Seven Replicates of Five Blocks

You can use PROC OPTEX to search for good resolvable block designs, but doing so requires a little trick. The DESIGN= option in the BLOCK statement enables you to specify a block structure that involves both replicates and blocks within replicates, but this is not enough. To see why, consider the following PROC OPTEX example, which sets up a candidate set of 15 treatments and a block structure data set that consists of seven replicates of five blocks of size 3 and then uses PROC OPTEX to find the best way to assign these 15 treatments to such a block structure, considering effects for both replicates and block within replicates.

The following DATA steps create two data sets: Candidates, which contains the candidate points for the design, and BlockStructure, which contains a block structure that involves both replicates and blocks within replicates.

```
data Candidates;
   do Treatment = 1 to 15;
      output;
   end;
run;
```

```
data BlockStructure;
   do Replicate = 1 to 7;
      do Block = 1 to 5;
         do Plot = 1 to 3;
            output;
         end;
      end;
   end;
run;
```

The following statements use PROC OPTEX to search for a good design for the candidate points that are stored in the data set Candidates by using the blocking structure that is contained in the data set BlockStructure:

```
proc optex data=Candidates coding=orthcan seed=492069001;
   class Treatment;
   model Treatment;
   blocks design=BlockStructure;
   class Replicate Block;
   model Replicate Block(Replicate);
   output out=Design;
   ods select BlockDesignEfficiencies;
run;
```

The DATA= option in the PROC OPTEX statement specifies that the data set Candidates contains the candidate points for the design. The SEED= option specifies an integer to use to start the pseudorandom number generator for initialization, guaranteeing reproducibility. The first CLASS statement refers to the data set Candidates and specifies **Treatment** as a classification variable. The first MODEL statement represents the treatment model; it specifies that the model consists of the main treatment effects. The DESIGN= option in the BLOCKS statement specifies that BlockStructure contains the fixed covariates for the model. In this case, the covariates are the replicates and the blocks. Because the second CLASS statement follows the BLOCKS statement, it refers to the data set BlockStructure and the model for the fixed covariates; it also specifies **Replicate** and **Block** as classification variables. Similarly, the MODEL statement that follows the BLOCKS statement refers to the data set BlockStructure and the model for the fixed covariates; it specifies fixed effects for **Replicate** and for **Block** nested within **Replicate**. The OUTPUT statement saves the best design in the data set Design, which is specified in the OUT= option.

Figure 5 shows that the best design found has a treatment D-efficiency of 71.43.

Figure 5 Example 3 OPTEX Output: Design Criteria (with No PRIOR= Option)

The OPTEX Procedure

Design Number	Treatment D-Efficiency	Treatment A-Efficiency
1	71.4286	71.4286
2	71.4286	71.4286
3	71.4286	71.4286
4	71.3371	71.2444
5	71.3371	71.2444
6	71.3371	71.2444
7	71.3371	71.2444
8	71.3371	71.2444
9	71.3371	71.2444
10	71.2004	70.9710

The following statement calls the %RBDEval macro, which evaluates the design:

```
%RBDEval(Design,Treatment,Replicate,Block);
```

The resulting efficiency measures for the two different blocking structures are shown in Figure 6. The design has a block design D-efficiency of 97.3 with respect to replicates. The design is not resolvable because it is not 100%

efficient with respect to replicates. However, with respect to blocks within replicates, the design has an efficiency of 100%, indicating that the design achieves balance in that sense.

Figure 6 Example 3 (No Prior) Macro Output: Design Efficiency Evaluation for Each Structure

Treatment Efficiency
Relative to Upper Bound for a Variance-Balanced Design

Evaluation	D-Efficiency
Replicate	97.3299
Block(Replicate)	100.0000

The following PROC FREQ step confirms that some treatments occur twice in some replicates and other treatments do not occur at all, as shown in Figure 7:

```
proc freq data=Design;
   table Treatment*Replicate / norow nocol nopct nocum;
run;
```

Figure 7 Treatment-by-Replicate Coincidence Counts for Optimal Figure 5 Design

The FREQ Procedure

Frequency	Table of Treatment by Replicate							
		Replicate						
Treatment	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Total**
1	1	0	0	2	2	1	1	7
2	2	1	1	0	1	1	1	7
3	1	1	0	2	1	1	1	7
4	0	2	1	1	1	1	1	7
5	1	0	2	1	0	0	3	7
6	2	1	2	0	1	1	0	7
7	1	1	1	0	2	1	1	7
8	0	2	1	1	1	1	1	7
9	1	1	1	1	1	1	1	7
10	2	0	1	1	1	1	1	7
11	0	2	1	1	1	1	1	7
12	2	1	1	1	1	1	0	7
13	0	1	2	1	1	1	1	7
14	1	1	1	1	0	2	1	7
15	1	1	0	2	1	1	1	7
Total	15	15	15	15	15	15	15	105

The reason why this optimal blocking approach fails to find a resolvable design is that PROC OPTEX can optimize only one definition of block D-efficiency, which is $|X'AX|$, at a time, and the blocking model that is used here defines the matrix A based only on the effect of blocks within replicates. Let A_B represent this matrix, and let A_R represent the corresponding matrix for replicates considered as blocks. Technically, A_R and A_B are defined as the projectors onto the residual spaces for the models that have effects only for replicates and only for blocks within replicates, respectively. To find a design that is both resolvable and balanced, you need to maximize the determinant of an information matrix that combines these two separate information matrices:

$$D^{\alpha} = |\alpha X' A_R X + (1-\alpha) X' A_B X|, \quad 0 < \alpha < 1$$

It can be shown that if a resolvable design exists, it maximizes such a criterion.

This is where the trick comes in. The PRIOR= option in the MODEL statement can be used with the block model and with the treatment model. For a block design whose blocks and replicates are of equal size, a block model of the form

```
class Replicate Block;
model Replicate, Block(Replicate) / prior=0,π;
```

defines a block D-efficiency of the form shown earlier, where

$$\alpha = \frac{\pi}{N + \pi}$$

where N is the size of the design. That is, a resolvable block design is Bayes optimal in the sense of DuMouchel and Jones (1994). The intuitive interpretation of this result is that, by claiming a certain amount of prior information about blocks within replicates, you free PROC OPTEX to try to find a design that contains information about replicates too.

The following statements exploit this method to find a resolvable balanced incomplete block design for 15 treatments in seven replicates of five blocks of size 3, by claiming about 11 observations' worth of prior information about the blocks-within-replicates effect:

```
proc optex data=Candidates coding=orthcan seed=215417959;
   class Treatment;
   model Treatment;
   blocks design=BlockStructure niter=10000 keep=10;
   class Replicate Block;
   model Replicate, Block(Replicate) / prior=0,11;
   output out=Design;
   ods select BlockDesignEfficiencies;
run;

%RBDEval(Design,Treatment,Replicate,Block);
```

Notice the high value of the NITER= option in the BLOCKS statement. Combinatorial designs of any sort can be difficult for PROC OPTEX to find, so the more iterations you allow, the better your odds of finding the theoretical optimum. On the other hand, PROC OPTEX invariably finds a pretty good design, if not the very best possible design. When you use the PRIOR= option, you must use commas to separate groups of effects in the MODEL statement that have the same prior precision. This example has two groups: **Replicate** and **Block(Replicate)**. The PRIOR= option specifies a prior precision value of 0 for **Replicate** and a prior precision value of 11 for **Block(Replicate)**.

Figure 8 shows that the best design found by using the prior information has a Bayesian treatment D-efficiency rating of 83.52.

Figure 8 Example 3 OPTEX Output: Bayesian Design Criteria (PRIOR=0,100)

The OPTEX Procedure

Design Number	Treatment D-Efficiency	Treatment A-Efficiency
1	83.5165	83.5165
2	83.5165	83.5165
3	83.4777	83.4388
4	83.4777	83.4388
5	83.4777	83.4388
6	83.4777	83.4388
7	83.4777	83.4388
8	83.4777	83.4388
9	83.4777	83.4388
10	83.4777	83.4388

Figure 9 shows that the design achieves a 100% block design D-efficiency with respect to replicates, indicating that

the design is resolvable. The design also achieves a 100% block design D-efficiency with respect to blocks within replicates, indicating that the design is balanced. Therefore, PROC OPTEX has provided a resolvable balanced incomplete block design.

Figure 9 Example 3 (with Prior) Macro Output: Design Efficiency Evaluation for Each Structure

Treatment Efficiency Relative to Upper Bound for a Variance-Balanced Design	
Evaluation	**D-Efficiency**
Replicate	100.0000
Block(Replicate)	100.0000

Value of π

A question that naturally arises when you use this approach is, "What value of π should you use for the nonzero prior on blocks within replicates?" Mathematically, a resolvable design is optimal for any value of π, but if you set it too small or too large, one or the other of the component information matrices will dominate the efficiency criterion. You want a prior high enough to ensure resolvability, but not so high that it compromises the ability of PROC OPTEX to find the most efficient resolvable design for the situation.

Currently, we have only limited empirical evidence, but $\pi = N/10$ has worked well so far. (We encourage feedback in this regard.) You could start with $\pi = N/10$ and NITER=10 to quickly assess resolvability. If the resulting design is not resolvable, then increase π (for example, double the value of the PRIOR= option) and try again. When the resulting design is resolvable, then you can decide whether or not to increase the value of the NITER= option (to 1,000 or 10,000, for example) to allow PROC OPTEX the opportunity to find a somewhat more efficient design. (Keep in mind that the time required will be approximately proportional to the value of the NITER= option). When PROC OPTEX finds a resolvable design, it is usually highly efficient compared to the most efficient possible resolvable design for the specific situation. Nevertheless, when $\pi = N/10$ results in resolvability, you can try reducing π (by half, for example) to see if resolvability is maintained and whether PROC OPTEX can find a resolvable design that has higher efficiency.

Randomization

Applying an experimental design in practice requires randomization: carefully scrambling the order of replicates, of blocks within replicates, and of plots within blocks. Randomization eradicates any potential hidden biases in how measurements are made, and in a sense it generates the appropriate analysis.

The randomized search that PROC OPTEX uses turns out designs whose rows are in some arbitrary order, conditioned by the search process. But "arbitrary" is not the same as "certifiably random." There are lots of ways to randomize block designs in SAS. A particularly convenient way is to use PROC PLAN with an input data set, as follows:

```
proc plan;
   factor Replicate = <number of reps>
          Block     = <number of blocks within reps>
          Plot      = <number of plots within blocks>;
   output out =RandomizedDesign
          data=UnrandomizedDesign;
run;
```

Unequally Sized Blocks

Block sizes do not need to be equal to use the technique involving the PRIOR= option in PROC OPTEX. In the wine example, if there had been 50 wines rather than 51, then you could have three sessions of sizes 17, 17, and 16. When block sizes are unequal, there is no theoretical upper bound for comparison, so the output from the %RBDEval macro changes.

Figure 10 Macro Output for Wine Example with Unequally Sized Blocks, Part 1

Wine Efficiency

Relative to Upper Bound for an Orthogonal Design

Evaluation	D-Efficiency	A-Efficiency
Subject	100.0000	100.0000
Session(Subject)	95.8849	95.8516

Figure 10 shows standard D-efficiency, which is 100% for subjects, indicating that the design is resolvable. Standard D- and A- efficiencies for sessions-within-subjects will always be less than 100% because the sessions are incomplete blocks. The 95.88% D-efficiency for sessions-within-subjects can be compared to the upper bound from similar designs with equal-sized blocks. In the first wine example, which has 51 treatments and 3 blocks of size 17, the upper bound is 96%, indicating that the design here with 95.88% efficiency is very efficient and cannot be substantially improved upon.

When block sizes are not equal, the output of the macro includes a second table, as shown in Figure 11. This table compares the standard efficiencies in the first table to the efficiency of the best design that PROC OPTEX can find when there is only a single block structure. This is a reasonable comparison because PROC OPTEX is known to find highly efficient designs for a single block structure. The design has a session-within-subjects efficiency of more than 99.9% relative to the best that PROC OPTEX can do for sessions within subjects, disregarding subjects.

Figure 11 Macro Output for Wine Example with Unequally Sized Blocks, Part 2

Wine Efficiency

Relative to Single Structure Search, niter=10

Evaluation	D-Efficiency	A-Efficiency
Subject	100.0000	100.0000
Session(Subject)	99.9777	99.9555

MORE GENERAL DESIGNS

You can use the Bayes optimality technique in PROC OPTEX to search for other designs that are efficient for more than one block structure. For example, consider a situation in which the natural block structure consists of days and shifts-within-days as follows: Each day, nine experimental units can be processed in three shifts of size 3. Although resolvable designs can effectively account for both day and shift effects, they are limited in this setting to studies that have nine treatments. Suppose the number of treatments of interest is six or 12 or some other number. You can use the PRIOR= option in PROC OPTEX to efficiently and simultaneously assign treatment combinations both to days (large blocks) and to sessions (nested blocks). You do not need to distort the natural block structure to fit the number of treatments, nor to distort the number of treatments to fit the natural block structure. This frees you to choose the best natural block structure and independently choose the number of treatments of interest and let PROC OPTEX do the combining in an efficient manner.

PROC OPTEX accepts whatever numbers and sizes of large or nested blocks are available for the study. As shown earlier, the nested blocks do not need to be the same size. Further, the large blocks can be unequal in size.

When the number of treatments is not equal to the size of the large blocks, you can no longer expect orthogonal blocking at that level. In many situations, leaving the prior π at the suggested $N/10$ can achieve good efficiency for both the large blocks and the nested blocks. If you want more efficiency for one structure at the expense of the other, you can vary π to accomplish that. Increasing π puts more emphasis on the large block structure.

APPENDIX: %RBDEVAL MACRO

The %RDBEval macro uses PROC OPTEX to evaluate an experimental design for efficiency with respect to replicates and blocks within replicates. When calling %RBDEval, you must specify as the first argument a SAS data set that contains the design to be evaluated, and you must specify three variables that identify the treatments, replicates, and blocks, respectively.

The syntax is as follows:

%RBDEval(*Design***,***vName***,***rName***,***bName***,NCHECK=,COMPARE=)**

You must specify the following required arguments:

Design
specifies a SAS data set that contains the design.

vName
specifies the variable that indexes the treatments.

rName
specifies the variable that indexes the replicates.

bName
specifies the variable that indexes the blocks.

You can also specify the following optional arguments:

COMPARE=BALANCED | SEARCH | ORTHOGONAL | ALL
specifies the relevant baseline for the relative efficiency values. This argument is critical. Efficiency is measured by some functional of a design's information matrix $X'X$. The most popular functional is the determinant of the information matrix $|X'X|$, which corresponds to the D-efficiency. But this value by itself is generally not very informative; you need to specify the COMPARE= argument to look at it relative to the best you can expect to achieve for the particular situation.

You can specify the following values for this argument:

BALANCED specifies that the relevant baselines are the theoretically optimal values for variance-balanced complete or incomplete block designs with respect to replicates and blocks, respectively, whether or not such variance-balanced designs actually exist for this block structure. This option does not apply if the block structure is not balanced.

If your block structure is balanced, with blocks of equal sizes and all replicates having the same number of blocks, then the theoretical optimal value of $|X'X|$ is known and makes a useful baseline for comparison. Therefore, COMPARE=BALANCED is the default in this case.

SEARCH specifies that the relevant baselines are the criterion values for the best designs found by searching for optimal replicate-only and blocks-only designs, respectively. By default, 10 searches are performed; you can change this with the NCHECK= argument.

If your block structure is not balanced, meaning that it has either unequally sized blocks or different numbers of blocks in replicates, then no optimal theoretical value for $|X'X|$ is known in general. In this case, your best bet is to search for an optimal replicates-only design to find a baseline value for replicate efficiency, and similarly an optimal blocks-only design for block efficiency. This form of comparison can be useful if balanced designs are not achievable for either replicates only or blocks only.

ORTHOGONAL requests that the standard D and A efficiency values be displayed. These efficiency values effectively use as the relevant baselines theoretically optimum values for orthogonal designs with respect to replicates and blocks, respectively, whether or not such designs actually exist for this block structure. For more information about how these values are computed, see the chapter "The OPTEX Procedure" in the *SAS/QC User's Guide*.

Because orthogonal designs are often not even nearly possible, especially for blocks-within-replicates, efficiencies relative to them will give you the least practical direct measure of how well your design behaves. However, such measures can be used to compare

designs that have different block structures. For example, for the unbalanced wine-tasting design discussed in the section on unequally sized blocks, you can use the COMPARE=ORTHOGONAL results to see that the corresponding efficiencies for the design that PROC OPTEX finds are very close to the known upper bounds on efficiencies for designs with similar but balanced block structures. Therefore, this output is included as part of the default for unbalanced block structures.

ALL specifies that efficiencies with respect to all appropriate baselines be displayed.

By default, COMPARE=BALANCED for designs with a balanced block structure, and COMPARE=ALL otherwise.

NCHECK=*number*

requests a search for optimal replicate-only and blocks-only designs, respectively, for the purpose of computing the relative efficiencies of a design; and specifies the number of times to repeat the search from different initial designs. By default, NCHECK=0, which indicates that no search is to be performed and that the relative efficiencies are not to be computed.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors:

Clifford Pereira
pereira@science.oregonstate.edu

Randy Tobias
Randy.Tobias@sas.com

ACKNOWLEDGMENT

Luna Sun of the Department of Statistics at Oregon State University provided valuable assistance with the practical review of the methodology.

REFERENCES

DuMouchel, W., and Jones, B. (1994). "A Simple Bayesian Modification of D-Optimal Designs to Reduce Dependence on an Assumed Model." *Technometrics* 36:37–47.

Kuehl, R. O. (2000). *Design of Experiments: Statistical Principles of Research Design and Analysis*. 2nd ed. Pacific Grove, CA: Brooks/Cole.

Morgan, J. P., and Reck, B. H. (2007). "Resolvable Designs with Large Blocks." *Annals of Statistics* 35:747–771.