The %MktBal Macro

Introduction

The %MktBal autocall macro creates factorial designs by using an algorithm that ensures that the design is perfectly balanced, or when the number of levels of a factor does not divide the number of runs, as close to perfectly balanced as possible. Before using the %MktBal macro, you should try using the %MktEx macro to see whether it makes a design that is balanced enough for your needs. The %MktEx macro can directly create thousands of orthogonal and balanced designs that the %MktBal macro will never find. Even when the %MktEx macro cannot create an orthogonal and balanced design, it usually finds a nearly balanced design. Designs that are created using the %MktBal macro, although perfectly balanced, might be less efficient than designs that are found using the %MktEx macro, and for large problems, the %MktBal macro can be slow.

The %MktBal macro is *not* a full-featured experimental design generator. For example, you cannot specify interactions that you want to estimate or specify restrictions such as which levels can or cannot appear together. You must use the %MktEx macro for that. The %MktBal macro builds a design by creating a balanced first factor, optimally (or nearly optimally) blocking it to create the second factor, then blocking the first two factors to create the third, and so on. After it creates all factors, it refines each factor. Each factor is in turn removed from the design, and the rest of the design is reblocked, replacing the initial factor if the new design is more *D*-efficient.

%MktBal Macro Syntax

%MktBal(*list*, **N=***n* <, *optional arguments*>**)**

Required Arguments

list

specifies a list of the numbers of levels of all the factors. For example, for three 2-level factors, specify either `2 2 2` or `2 ** 3`. You can specify lists of numbers, such as `2 2 3 3 4 4`, or you can use a *levels***number-of-factors* syntax, such as `2**2 3**2 4**2`. The specification `3**2` means two 3-level factors. You can also use combinations such as `2 2 3**4 5 6`. The factor list is a positional argument. This means that it must come first, and unlike all other arguments, it is not specified after a name and an equal sign.

N=*n*

specifies the number of runs in the design. You can use the %MktRuns macro to get suggestions for values of N=.

Optional Arguments

INIT=*SAS-data set*

specifies a SAS data set that contains the initial design. You can specify just the first few columns of the design in this data set by using the column names X1, X2, and so on, and these columns will never change.

NOTE: This is different from the INIT= data set in the %MktEx macro. It is often the case that the first few columns can be constructed combinatorially, and it is known that the optimal design will add only new columns to the initial few orthogonal columns. This argument makes that process more efficient. By default, when you do not specify INIT=, the %MktBal macro iteratively finds and improves all columns.

MAXINITITER=*n*

specifies the maximum number of random starts for each factor during the initialization stage. This is the number of iterations that PROC OPTEX performs for each factor during the initialization. With larger values of *n*, the macro tends to find slightly better designs at a cost of slower run times. The default is the value of the MAXSTARTS= argument.

MAXITER=*n*

ITER=*n*

specifies the maximum number of iterations (designs to create). By default, MAXITER=5. This is the outermost set of iterations.

MAXSTARTS=*n*

specifies the maximum number of random starts for each factor. This is the number of iterations that PROC OPTEX performs for each factor. With larger values of *n*, the macro tends to find slightly better designs at a cost of slower run times. By default, MAXSTARTS=10.

MAXTRIES=*n*

specifies the maximum number of times to try refining each factor after the initialization stage. By default, MAXTRIES=10. Increasing the value of *n* usually has little effect, because the macro stops refining each design when the efficiency stabilizes.

OPTIONS=*options-list*

specifies binary arguments. You can specify the following:

NOPRINT

requests that the final *D*-efficiency not be displayed.

NOSORT

requests that the final design not be sorted.

OA

searches for an orthogonal array. Factors are added sequentially, and they are acceptable only when *D*-efficiency reaches 100. You can use this argument together with the %MktBal macro to search for small orthogonal arrays, or in some cases to augment existing orthogonal arrays.

PROGRESS

reports on the macro's progress, giving you information about which step is being executed. For large numbers of factors or a large number or runs, or when the number of levels is large, this macro is slow.

SEQUENTIAL
sequentially adds factors to the design and does not go back and refine them.

By default, none of these arguments are specified.

OUT=*SAS-data-set*
specifies the name of the output data set that contains the experimental design. By default, OUT=Design.

SEED=*n*
specifies the random number seed. By default, SEED=0, and clock time is used to make the random number seed.

Help Option

You can specify either of the following to display the option names and simple examples of the macro syntax:

```
%mktbal(help)
%mktbal(?)
```

%MktBal Macro Notes

This macro specifies **options nonotes** throughout most of its execution. If you want to see all the notes, submit the following statement before running the macro:

```
%let mktopts = notes;
```

To see the macro version, submit the following statement before running the macro:

```
%let mktopts = version;
```

Example

Suppose you want to create a balanced design that contains two 2-level factors and three 3-level factors in 18 runs. The following SAS statement invokes the %MktBal macro to create the design:

```
%mktbal(2 2 3 3 3, n=18, seed=151)
```

Figure 1 displays the results. The design that the %MktBal macro generates has a D-efficiency of 99.86 and an average prediction standard error of 0.71. This design is optimal. A 100% efficient design does not exist, because there are two 2-level factors and 18 is not divisible by $2 \times 2 = 4$.

Figure 1 Design Efficiency Measures

The OPTEX Procedure

Design Number	D-Efficiency	A-Efficiency	G-Efficiency	Average Prediction Standard Error
1	99.8621	99.7230	98.6394	0.7071

The following statements print the resulting design matrix:

```
proc print data=design noobs;
run;
```

Figure 2 displays the resulting design matrix, which is saved in the data set Design. By default, the factors in the output data set Design are named x1, x2, x3, and so on. You can use the %MktLab macro to conveniently change them.

Figure 2 Design Matrix

x1	x2	x3	x4	x5
1	1	1	1	1
1	1	1	3	3
1	1	2	1	2
1	1	2	2	3
1	2	1	2	2
1	2	2	2	1
1	2	3	1	3
1	2	3	3	1
1	2	3	3	2
2	1	1	3	1
2	1	2	3	2
2	1	3	1	3
2	1	3	2	1
2	1	3	2	2
2	2	1	1	2
2	2	1	2	3
2	2	2	1	1
2	2	2	3	3