%MktDups Macro

The `%MktDups` autocall macro detects duplicate choice sets and duplicate alternatives within generic choice sets. See the following pages for examples of using this macro in the design chapter: 147, 174, 198, and 206. Also see the following pages for examples of using this macro in the discrete choice chapter: 319, 368, 519, 564, 564, 567, 570, 576, 597, 607, 617, 628, 636, 645, 650, 654, 656, 659 and 662. Additional examples appear throughout this chapter. To illustrate, consider a simple experiment with these two choice sets. These choice sets are completely different and are not duplicates.

```
a b c       a b c
1 2 1       1 1 1
2 1 2       2 2 2
1 1 2       2 2 1
2 1 1       1 2 2
```

Now consider these two choice sets:

```
a b c       a b c
1 2 1       2 1 2
2 1 2       1 1 2
1 1 2       2 1 1
2 1 1       1 2 1
```

They are the same for a generic study because all of the same alternatives are there, they are just in a different order. However, for a branded study they are different. For a branded study, there would be a different brand for each alternative, so the choice sets would be the same only if all the same alternatives appeared in the same order. For both a branded and generic study, these choice sets are duplicates:

```
a b c       a b c
1 2 1       1 2 1
2 1 2       2 1 2
1 1 2       1 1 2
2 1 1       2 1 1
```

Now consider these choice sets for a generic study.

```
a b c       a b c
1 2 1       1 2 1
2 1 1       1 2 1
1 1 2       1 1 2
2 1 1       2 1 1
```

First, each of these choice sets has duplicate alternatives (2 1 1 in the first and 1 2 1 in the second). Second, these two choice sets are flagged as duplicates, even though they are not exactly the same. They are flagged as duplicates because every alternative in choice set one is also in choice set two, and every alternative in choice set two is also in choice set one. In generic studies, two choice sets are considered duplicates unless one has one or more alternatives that are not in the other choice set.

As an example, a design is created with the `%ChoicEff` macro choice-set-swapping algorithm for a branded study, then the `%MktDups` macro is run to check for and eliminate duplicate choice sets. The following steps create and evaluate the design:

```
%mktex(3 ** 9, n=27, seed=424)

data key;
   input (Brand x1-x3) ($);
   datalines;
Acme   x1 x2 x3
Ajax   x4 x5 x6
Widgit x7 x8 x9
;

%mktroll(design=randomized, key=key, alt=brand, out=cand)

%choiceff(data=cand,                /* candidate set of choice sets         */
          model=class(brand x1-x3 / sta),/* model with stdz orthog coding   */
          seed=420,                 /* random number seed                   */
          nsets=18,                 /* number of choice sets                */
          nalts=3,                  /* number of alternatives               */
          options=relative,         /* display relative D-efficiency        */
          beta=zero)                /* assumed beta vector, Ho: b=0         */

proc freq; tables set; run;

%mktdups(branded, data=best, factors=brand x1-x3, nalts=3, out=out)

proc freq; tables set; run;
```

The following PROC FREQ results show that candidate choice sets occur more than once in the design:

The FREQ Procedure

Set	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	3	5.56	3	5.56
2	6	11.11	9	16.67
3	3	5.56	12	22.22
5	6	11.11	18	33.33
8	3	5.56	21	38.89
9	3	5.56	24	44.44
12	3	5.56	27	50.00
13	3	5.56	30	55.56
14	6	11.11	36	66.67

16	6	11.11	42	77.78
19	3	5.56	45	83.33
20	3	5.56	48	88.89
24	3	5.56	51	94.44
27	3	5.56	54	100.00

The %MktDups macro displays the following information in the log:

```
Design:          Branded
Factors:         brand x1-x3
                 Brand
                 x1 x2 x3
Duplicate Sets:  4
```

The output from the %MktDups macro contains the following table:

Choice Set	Duplicate Choice Sets To Delete
1	15
3	18
7	14
10	17

The first line of the first table tells us that this is a branded design as opposed to generic. The second line tells us the factors as specified in the factors= option. These are followed by the actual variable names for the factors. The last line reports the number of duplicates. The second table tells us that choice set 1 is the same as choice set 15. Similarly, 3 and 18 are the same, and so on. The out= data set will contain the design with the duplicate choice set eliminated.

Now consider an example with purely generic alternatives. The following steps create and evaluate the design:

```
%mktex(2 ** 5, n=2**5, seed=109)

%choiceff(data=randomized,          /* candidate set of alternatives        */
          model=class(x1-x5 / sta), /* model with stdz orthogonal coding    */
          seed=93,                  /* random number seed                   */
          nsets=42,                 /* number of choice sets                */
          flags=4,                  /* 4 alternatives, generic candidates   */
          options=relative,         /* display relative D-efficiency        */
          beta=zero)                /* assumed beta vector, Ho: b=0         */

%mktdups(generic, data=best, factors=x1-x5, nalts=4, out=out)
```

The macro produces the following tables:

```
Design:          Generic
Factors:         x1-x5
                 x1 x2 x3 x4 x5
Sets w Dup Alts: 1
Duplicate Sets:  1
```

Choice Set	Duplicate Choice Sets To Delete	
2	25	
39		Alternatives

For each choice set listed in the choice set column, either the other choice sets it duplicates are listed or the word `Alternatives` is displayed if the problem is with duplicate alternatives.

The following step displays just the choice sets with duplication problems:

```
proc print data=best;
   var x1-x5;
   id set; by set;
   where set in (2, 25, 39);
   run;
```

The results are as follows:

Set	x1	x2	x3	x4	x5
2	1	2	1	1	1
	2	2	1	1	1
	1	1	2	2	2
	2	1	2	2	2
25	1	1	2	2	2
	2	1	2	2	2
	2	2	1	1	1
	1	2	1	1	1
39	1	1	2	1	1
	1	1	2	1	1
	2	2	1	2	2
	2	2	1	2	2

You can see that the macro detects duplicates even though the alternatives do not always appear in the same order in the different choice sets.

Now consider another example. The following steps create and evaluate a choice design:

```
%mktex(2 ** 6, n=2**6)

%mktroll(design=design, key=3 2, out=cand)

%mktdups(generic, data=cand, factors=x1-x2, nalts=3, out=out)

proc print; by set; id set; run;
```

Some of the results are as follows:

```
Design:          Generic
Factors:         x1-x2
                 x1 x2
Sets w Dup Alts: 40
Duplicate Sets:  50
```

Choice Set	Duplicate Choice Sets To Delete	
1		Alternatives
2		Alternatives
	5	
	6	
	17	
	18	
	21	
.		
.		
.		

The output lists, for each set of duplicates, the choice set that is kept (in the first column) and all the matching choice sets that are deleted (in the second column).

The unique choice sets are as follows:

Set	_Alt_	x1	x2
7	1	1	1
	2	1	2
	3	2	1
8	1	1	1
	2	1	2
	3	2	2

```
12      1       1       1
        2       2       1
        3       2       2

28      1       1       2
        2       2       1
        3       2       2
```

The following example creates a conjoint design* and tests it for duplicates:

```
%mktex(3 ** 3 2 ** 2, n=19, seed=513)

%mktdups(linear, data=design, factors=x1-x5)
```

The results are as follows:

```
Design:          Linear
Factors:         x1-x5
                 x1 x2 x3 x4 x5
Duplicate Runs:  2
```

Run	Duplicate Runs To Delete
12	13
16	17

%MktDups Macro Options

The following options can be used with the `%MktDups` macro:

Option	Description
`options`	(positional) binary options
	(positional) "help" or "?" displays syntax summary
`data=`*SAS-data-set*	input choice design
`factors=`*variable-list*	factors in the design
`nalts=`*n*	number of alternatives
`out=`*SAS-data-set*	output data set
`outlist=`*SAS-data-set*	output data set with duplicates
`vars=`*variable-list*	factors in the design

*Normally, we would use 18 runs and not a prime number like 19 that is not divisible by any of the numbers of levels, 2 and 3. We picked a silly number like 19 to ensure duplicates for this example.

Help Option

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

```
%mktdups(help)
%mktdups(?)
```

Positional Parameter

The options list is a positional parameter. This means it must come first, and unlike all other parameters, it is not specified after a name and an equal sign.

options

specifies positional options. You can specify `noprint` and one of the following: `generic`, `branded`, or `linear`.

`branded`
specifies that since one of the factors is brand, the macro only needs to compare corresponding alternatives in each choice set.

`generic`
specifies a generic design and is the default. This means that there are no brands, so options are interchangeable, so the macro needs to compare each alternative with every other alternative in every choice set.

`linear`
specifies a linear not a choice design. Specify `linear` for a full-profile conjoint design, for an ANOVA design, or for the linear version of a branded choice design.

`noprint`
suppresses the output display. This option is used when you are only interested in the output data set or macro variable.

The following step shows an example of the `noprint` option:

```
%mktdups(branded noprint, data=design, nalts=3)
```

Required Options

This next option is mandatory with choice designs.

nalts= *n*

specifies the number of alternatives. This option must be specified with generic or branded choice designs. It is ignored with linear model designs. For generic or branded designs, the `data=` data set must contain `nalts=` observations for the first choice set, `nalts=` observations for the second choice set, and so on.

Other Options

The other options are as follows:

data= *SAS-data-set*
specifies the input choice design. By default, the macro uses the last data set created.

out= *SAS-data-set*
specifies an output data set that contains the design with duplicate choice sets excluded. By default, no data set is created, and the macro just reports on duplicates. Often, you will want to specify a two-level name to create a permanent SAS data set so the design is available later for analysis.

outlist= *SAS-data-set*
specifies the output data set with the list of duplicates. By default, `outlist=outdups`.

vars= *variable-list*
factors= *variable-list*
specifies the factors in the design. By default, all numeric variables are used.

%MktDups Macro Notes

This macro specifies `options nonotes` throughout most of its execution. If you want to see all of the notes, submit the statement `%let mktopts = notes;` before running the macro. To see the macro version, submit the statement `%let mktopts = version;` before running the macro.