%MktRoll Macro

The `%MktRoll` autocall macro constructs a choice design from a linear arrangement. See the following pages for examples of using the `%MktRoll` macro in the design chapter: 134 and 192. Also see the following pages for examples of using this macro in the discrete choice chapter: 312, 320, 357, 387, 429, 505, 546, 556, 575, 607, 617, 628 and 636. Additional examples appear throughout this chapter. The `%MktRoll` macro takes as input a SAS data set containing an experimental design with one row per choice set, the *linear arrangement*, for example, a design created by the `%MktEx` macro. This data set is specified in the `design=` option. This data set has one variable for each attribute of each alternative in the choice experiment. The output from this macro is an `out=` SAS data set is the *choice design* containing the experimental design with one row per alternative per choice set. There is one column for each different attribute. For example, in a simple branded study, `design=` could contain the variables `x1-x5` which contain the prices of each of five alternative brands. The output data set would have one factor, `Price`, that contains the price of each of the five alternatives. In addition, it would have the number (or optionally the name) of each alternative.[*]

The rules for determining the mapping between factors in the `design=` data set and the `out=` data set are contained in the `key=` data set. For example, assume that the `design=` data set contains the variables `x1-x5` which contain the prices of each of five alternative brands: Brand A, B, C, D, and E. The choice design has two factors, `Brand` and `Price`. Brand A price is made from `x1`, Brand B price is made from `x2`, ..., and Brand E price is made from `x5`. A convenient way to get all the names in a variable list like `x1-x5` is with the `%MktKey` macro. The following step creates the five names in a single column:

```
%mktkey(5 1)
```

The `%MktKey` macro produces the following data set:

```
                    x1

                    x1
                    x2
                    x3
                    x4
                    x5
```

The following step creates the `Key` data set:

[*]See page 67 for an explanation of the linear arrangement of a choice design versus the arrangement of a choice design that is more suitable for analysis.

```
data key;
   input (Brand Price) ($);
   datalines;
A x1
B x2
C x3
D x4
E x5
;
```

This data set has two variables. `Brand` contains the brand names, and `Price` contains the names of the factors that are used to make the price effects for each of the alternatives. The `out=` data set will contain the variables with the same names as the variables in the `key=` data set.

The following step creates the linear arrangement with one row per choice set:

```
%mktex(3 ** 5, n=12)
```

The following step creates the choice design with one row per alternative per choice set:

```
%mktroll(design=randomized, key=key, out=sasuser.design, alt=brand)
```

Consider, for example, a randomized data set that contains the following row:

```
Obs    x1    x2    x3    x4    x5

  9     3     1     1     2     1
```

Then the data set SASUSER.DESIGN contains the following rows:

```
Obs    Set    Brand    Price

 41     9       A        3
 42     9       B        1
 43     9       C        1
 44     9       D        2
 45     9       E        1
```

The price for Brand A is made from `x1=3`, ..., and the price for Brand E is made from `x5=1`.

Now assume that there are three alternatives, each a different brand, and each composed of four factors: `Price`, `Size`, `Color`, and `Shape`. In addition, there is a constant alternative. First, the `%MktEx` macro is used to create a design with 12 factors, one for each attribute of each alternative. The following step creates the design:

```
%mktex(2 ** 12, n=16, seed=109)
```

The following step creates the `key=` data set:

```
data key;
   input (Brand Price Size Color Shape) ($); datalines;
          A     x1     x2   x3    x4
          B     x5     x6   x7    x8
          C     x9    x10  x11   x12
          None   .      .    .     .
;
```

It shows that there are three brands, A, B, and C, and also None.

Brand A is created from `Brand` = “A”, `Price` = `x1`, `Size` = `x2`, `Color` = `x3`, `Shape` = `x4`.

Brand B is created from `Brand` = “B”, `Price` = `x5`, `Size` = `x6`, `Color` = `x7`, `Shape` = `x8`.

Brand C is created from `Brand` = “C”, `Price` = `x9`, `Size` = `x10`, `Color` = `x11`, `Shape` = `x12`.

The constant alternative is created from `Brand` = “None” and none of the attributes. The “`.`” notation is used to indicate missing values in input data sets. The actual values in the `Key` data set are blank (character missing).

The following step creates the design with one row per alternative per choice set:

```
%mktroll(design=randomized, key=key, out=sasuser.design, alt=brand)
```

Consider, for example, a randomized data set that contains the following row:

```
 Obs    x1    x2    x3    x4    x5    x6    x7    x8    x9   x10    x11    x12

   8     2     2     2     2     2     1     1     2     2     2      1      2
```

Then the data set SASUSER.DESIGN contains the following rows:

```
              29      8    A          2        2       2        2
              30      8    B          2        1       1        2
              31      8    C          2        2       1        2
              32      8    None       .        .       .        .
```

Now assume like before that there are three branded alternatives, each composed of four factors: `Price`, `Size`, `Color`, and `Shape`. In addition, there is a constant alternative. Also, there is an alternative-specific factor, `Pattern`, that only applies to Brand A and Brand C. First, the `%MktEx` macro is used to create a design with 14 factors, one for each attribute of each alternative. The following step creates the design:

```
%mktex(2 ** 14, n=16, seed=114)
```

The following step creates the key= data set:

```
data key;
   input (Brand Price Size Color Shape Pattern) ($);
   datalines;
A     x1     x2   x3    x4   x13
B     x5     x6   x7    x8     .
C     x9    x10  x11   x12   x14
None   .      .    .     .     .
;
```

It shows that there are three brands, A, B, and C, plus None.

Brand A is created from Brand = "A", Price = x1, Size = x2, Color = x3, Shape = x4, Pattern = x13.

Brand B is created from Brand = "B", Price = x5, Size = x6, Color = x7, Shape = x8.

Brand C is created from Brand = "C", Price = x9, Size = x10, Color = x11, Shape = x12, Pattern = x14.

The constant alternative is Brand = "None" and none of the attributes.

The following step creates the design with one row per alternative per choice set:

```
%mktroll(design=randomized, key=key, out=sasuser.design, alt=brand)
```

Consider, for example, a randomized data set that contains the following row:

```
 Obs   x1   x2   x3   x4   x5   x6   x7   x8   x9   x10   x11   x12   x13   x14

   8    2    1    1    2    1    2    1    2    1    1     1     2     1     2
```

Then the data set SASUSER.DESIGN contains the following rows:

```
Obs    Set    Brand    Price    Size    Color    Shape    Pattern

 29      8    A          2        1       1        2         1
 30      8    B          1        2       1        2         .
 31      8    C          1        1       1        2         2
 32      8    None       .        .       .        .         .
```

Now assume we are going to fit a model with price cross-effects so we need x1, x5, and x9 (the three price effects) available in the out= data set. See pages 444 and 468 for other examples of cross-effects. The following step creates the design:

```
%mktroll(design=randomized, key=key, out=sasuser.design, alt=brand,
         keep=x1 x5 x9)
```

Now the data set also contains the three original price variables, for example, as follows:

```
 Obs    Set    Brand    Price    Size    Color    Shape    Pattern    x1    x5    x9

  29     8     A          2        1       1        2         1        2     1     1
  30     8     B          1        2       1        2         .        2     1     1
  31     8     C          1        1       1        2         2        2     1     1
  32     8     None       .        .       .        .         .        2     1     1
```

Every value in the `key=` data set must appear as a variable in the `design=` data set. The macro displays a warning if it encounters a variable name in the `design=` data set that does not appear as a value in the `key=` data set.

%MktRoll Macro Options

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

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`alt=`*variable*	variable with name of each alternative
`design=`*SAS-data-set*	input SAS data set
`keep=`*variable-list*	factors to keep
`key=`*SAS-data-set*	`Key` data set name
`key=`*rows columns* <`t`>	`Key` data set description
`options=nowarn`	suppress the variables not mentioned warning
`out=`*SAS-data-set*	output SAS data set
`set=`*variable*	choice set number variable

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

```
%mktroll(help)
%mktroll(?)
```

You must specify the `design=` and `key=` options.

alt= *variable*

specifies the variable in the `key=` data set that contains the name of each alternative. Often this is something like `alt=Brand`. When `alt=` is not specified, the macro creates a variable `_Alt_` that contains the alternative number.

design= *SAS-data-set*

specifies an input SAS data set with one row per choice set. The `design=` option must be specified.

keep= *variable-list*
specifies factors from the `design=` data set that should also be kept in the `out=` data set. This option is useful to keep terms that are used to create cross-effects.

key= *SAS-data-set* | *l*ist
specifies the rules for mapping the `design=` data set to the `out=` data set. The `key=` option must be specified. It has one of two forms. 1) The `key=` option names an input SAS data set containing the rules for mapping the `design=` data set to the `out=` data set. The structure of this data set is described in detail in the preceding examples. 2) When you want the `key=` data set to exactly match the data set that comes out of the MktKey macro, you can specify the argument to the MktKey macro directly in the `key=` option, and the `%MktRoll` macro will make the `key=key` data set for you. In other words, the following two specifications are equivalent:

```
%mktkey(3 3 t)
%mktroll(design=design, key=key,   out=frac)

%mktroll(design=design, key=3 3 t, out=frac)
```

options= *options-list*
specifies binary options. By default, none of these options are specified. Specify one or more of the following values after `options=`.

`nowarn`
does not display a warning when the `design=` data set contains variables not mentioned in the `key=` data set. Sometimes this is perfectly fine.

out= *SAS-data-set*
specifies the output SAS data set. If `out=` is not specified, the DATAn convention is used.

set= *variable*
specifies the variable in the `out=` data set that will contain the choice set number. By default, this variable is named `Set`.

%MktRoll 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.