ChoiceEff Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`bestcov=`*SAS-data-set*	covariance matrix for the best design
`bestout=`*SAS-data-set*	best design
`beta=`*list*	true parameters
`chunks=`*n*	number of observations to code at once
`converge=`*n*	convergence criterion
`cov=`*SAS-data-set*	all of the covariance matrices
`data=`*SAS-data-set*	input choice candidate set
`drop=`*variable-list*	variables to drop from the model
`fixed=`*variable-list*	variable that flags fixed alternatives
`flags=`*variable-list*|*n*	variables that flag the alternatives
`init=`*SAS-data-set*	input initial design data set
`initvars=`*variable-list*	initial variables
`intiter=`*n*	maximum number of internal iterations
`iter=`*n*	maximum iterations (designs to create)
`maxiter=`*n*	maximum iterations (designs to create)
`model=`*model-specification*	`model` statement list of effects
`morevars=`*variable-list*	more variables to add to the model
`n=`*n*	number of observations
`nalts=`*n*	number of alternatives
`nsets=`*n*	number of choice sets desired
`options=coded`	displays the coded candidate set
`options=detail`	displays the details of the swaps
`options=nobeststar`	no asterisk when a better design is found
`options=nocode`	skips the PROC TRANSREG coding stage
`options=nodups`	prevents duplicate choice set creation
`options=notests`	suppress the hypothesis tests
`options=orthcan`	orthogonalizes the candidate set
`options=outputall`	outputs all designs to OUT= and COV=
`options=relative`	displays final relative *D*-efficiency
`options=resrep`	same as `options=detail`
`out=`*SAS-data-set*	all designs data set
`restrictions=`*macro-name*	restrictions macro
`resvars=`*variable-list*	variables for restrictions
`rscale=`*r*	relative efficiency scaling factor
`rscale=generic`	equivalent to `rscale=n` (*n* sets)
`rscale=alt`	for simple alternative-specific designs
`rscale=partial=`*p* `of` *q*	*p* of *q* alternatives or attributes vary
`seed=`*n*	random number seed
`submat=`*number-list*	submatrix for efficiency calculations
`types=`*integer-list*	number of sets of each type
`typevar=`*variable*	choice set types variable
`weight=`*weight-variable*	optional weight variable

```
%mktex(help) /* show option list with ? or help */


%mktex(3 ** 3, n = 3**3) /* candidate set of alternatives */

%choiceff(data=design, model=class(x1-x3 / sta), nsets=9, options=relative,
          maxiter=10, seed=104, flags=3, beta=zero)

%mktex(3 ** 9, n=2187) /* candidate set of choice sets */

%mktroll(design=randomized, key=3 3, out=frac)

%choiceff(data=frac, model=class(x1-x3 / sta), seed=104, maxiter=10,
          options=relative, nsets=9, nalts=3, beta=zero)


%mktex(3 ** 4, n = 9) /* design evaluation */

%mktlab(data=design, vars=Set x1-x3)

%choiceff(data=final, model=class(x1-x3 / sta), nsets=3, nalts=3,
          options=relative, beta=zero, init=final(keep=set), intiter=0)


%mktex(4 ** 6, n=32, seed=104)      /* choice design with restrictions      */

%macro res;
   do i = 1 to nalts;
      do k = i + 1 to nalts;
         if all(x[i,] >= x[k,])     /* alt i dominates alt k                */
            then bad = bad + 1;
         if all(x[k,] >= x[i,])     /* alt k dominates alt i                */
            then bad = bad + 1;
         end;
      end;
   %mend;

%choiceff(data=randomized,          /* candidate set of alternatives        */
          model=class(x1-x6 / sta), /* model with stdz orthogonal coding    */
          nsets=8,                  /* number of choice sets                */
          flags=4,                  /* 4 alternatives, generic candidates   */
          seed=104,                 /* random number seed                   */
          options=relative          /* display relative D-efficiency        */
                  resrep,           /* detailed report on restrictions      */
          restrictions=res,         /* name of the restrictions macro       */
          resvars=x1-x6,            /* variable names used in restrictions  */
          maxiter=1,                /* maximum number of designs to make    */
          bestout=desres1,          /* final choice design                  */
          beta=zero)                /* assumed beta vector, Ho: b=0         */
```

MktAllo Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`data=`*SAS-data-set*	input SAS data set
`freq=`*variable*	frequency variable
`nalts=`*n*	number of alternatives
`out=`*SAS-data-set*	output SAS data set
`vars=`*variable-list*	input variables

```
%mktallo(help) /* show option list with ? or help */

data allocs2;
   input Brand $ 1-8 Price $ Count Set;
   datalines;
.              .         0 1
Brand  1     $50       103 1
Brand  2     $75        58 1
Brand  3     $50       318 1
Brand  4    $100        99 1
Brand  5    $100        54 1
Brand  6    $100        83 1
Brand  7     $75        71 1
Brand  8     $75        58 1
Brand  9     $75       100 1
Brand 10     $50        56 1
.              .        10 2
Brand  1    $100        73 2
Brand  2    $100        76 2
Brand  3    $100       342 2
Brand  4     $50        55 2
Brand  5     $75        50 2
Brand  6    $100        77 2
Brand  7     $75        95 2
Brand  8    $100        71 2
Brand  9     $50        72 2
Brand 10    $100        79 2
;

%mktallo(data=allocs2, out=allocs3, nalts=11,
         vars=set brand price, freq=Count)
```

MktBal Macro Options

Option	Description
`list`	(positional) list of the numbers of levels
	(positional) "help" or "?" displays syntax summary
`init=`*SAS-data-set*	initial input experimental design
`iter=`*n*	maximum iterations (designs to create)
`maxinititer=`*n*	maximum initialization iterations
`maxiter=`*n*	maximum iterations (designs to create)
`maxstarts=`*n*	maximum number of random starts
`maxtries=`*n*	times to try refining each factor
`n=`*n*	number of runs in the design
`options=noprint`	suppress the final *D*-efficiency
`options=nosort`	do not sort the final design
`options=oa`	an orthogonal array is sought
`options=progress`	reports on macro progress
`options=sequential`	factors are added but not refined
`out=`*SAS-data set*	output experimental design
`seed=`*n*	random number seed

```
%mktbal(help) /* show option list with ? or help */


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


%mktbal(2 ** 23, n=24, options=oa progress, maxstarts=2000, seed=104)


%mktex(10 ** 4, n=100)
%mktbal(10 ** 4 2 2, n=100, init=design, options=oa progress,
        maxstarts=5000, seed=104, out=balanced)
```

MktBIBD Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`b=`*b*	number of blocks (alias for `nsets=`)
`group=`*n*	number of groups
`k=`*k*	block size (alias for `setsize=`)
`nattrs=`*t*	number of attributes (alias for `t=`)
`nsets=`*b*	number of sets (alias for `b=`)
`optiter=`*n1 < n2 < n3 >>*	number of PROC OPTEX iterations
`options=position`	optimizes position frequencies
`options=neighbor`	nondirectional row-neighbor balance and position
`options=serial`	directional row-neighbor balance and position
`out=`*SAS-data-set*	output data set BIBD
`outf=`*SAS-data-set*	output data set factorial design
`outi=`*SAS-data-set*	output data set incidence matrix
`outs=`*SAS-data-set*	output data set sorted design
`positer=`*n1 < n2 < n3 >>*	number of iterations position frequencies
`seed=`*n*	random number seed
`setsize=`*k*	set size (alias for `k=`)
`t=`*t*	number of treatments (alias for `nattrs=`)
`weights=`*n n*	position frequency badness weights

```
%mktbibd(help) /* show option list with ? or help */


%mktbibd(b=10, t=5, k=3, seed=104)


%mktbibd(nsets=10, nattrs=5, setsize=3, seed=104)


%mktbibd(b=14, t=7, k=4, options=neighbor, seed=104)


%mktbibd(b=14, t=7, k=4, options=serial, seed=361699)
```

MktBlock Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`alt=`*variable*	alternative number variable
`block=`*variable*	block number variable
`data=`*SAS-data-set*	either the choice or linear model design
`factors=`*variable-list*	factors in the design
`id=`*variable-list*	variables to copy to output data set
`initblock=`*variable*	initial blocking variable
`iter=`*n*	times to try to block the design
`list=`*n*	list larger canonical correlations
`maxiter=`*n*	times to try to block the design
`nalts=`*n*	number of alternatives in choice set
`nblocks=`*n*	number of blocks to create
`next=`*n*	where to look for the next exchange
`options=nosort`	do not sort the design into blocks
`out=`*SAS-data-set*	output data set with block numbers
`outr=`*SAS-data-set*	randomized output data set
`print=`*print-options*	output display options
`ridge=`*n*	ridging factor
`seed=`*n*	random number seed
`set=`*variable*	choice set number variable
`vars=`*variable-list*	factors in the design

```
%mktblock(help) /* show option list with ? or help */


%mktex(3 ** 6, n=27)

%mktblock(data=randomized, nblocks=3, factors=x1-x6, seed=377)


%mktex(3 ** 6, n=729)

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

%choiceff(data=out, model=class(x1-x3), nsets=18, nalts=2,
          beta=zero, options=nodups, seed=151)

%mktblock(data=best, nalts=2, nblocks=2, factors=x1-x3, seed=472)
```

MktBSize Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`b=`*do-list*	number of blocks (alias for `nsets=`)
`k=`*do-list*	block size (alias for `setsize=`)
`maxreps=`*n*	maximum number of replications
`nattrs=`*do-list*	number of attributes (alias for `t=`)
`nsets=`*do-list*	number of sets (alias for `b=`)
`order=`*order-list*	order of the loops
`options=nocheck`	suppress checking b, t, and k
`options=ubd`	lifts the balance restriction on the design
`out=`*SAS-data-set*	output data set design list
`setsize=`*do-list*	set size (alias for `k=`)
`t=`*do-list*	number of treatments (alias for `nattrs=`)

```
%mktbsize(help) /* show option list with ? or help */

%mktbsize(t=12, k=4 to 8, b=12 to 30)

%mktbsize(nattrs=12, setsize=4 to 8, nsets=12 to 30)

%mktbsize(t=5 to 20, k=3 to t - 1, nsets=t to 30)

%mktbsize(nattrs=5 to 20, setsize=3 to t - 1, nsets=t to 30)

%mktbsize(t=20, k=6, options=ubd)

%mktbsize(nattrs=20, setsize=6, options=ubd)
```

MktDes Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`big=`*n*	size of big candidate set
`cand=`*SAS-data-set*	candidate design
`classopts=`*options*	`class` statement options
`coding=`*name*	`coding=` option
`examine=`< I > < V >	matrices that you want to examine
`facopts=`*options*	PROC FACTEX statement options
`factors=`*factor-list*	factors and levels for each factor
`generate=`*options*	`generate` statement options
`interact=`*interaction-list*	interaction terms
`iter=`*n*	number of designs
`keep=`*n*	number of designs to keep
`maxiter=`*n*	number of designs
`method=`*name*	search method
`n=`*n* | SATURATED	number of runs
`nlev=`*n*	number of levels for pseudo-factors
`options=allcode`	shows all code
`options=check`	checks the `cand=` design
`options=nocode`	suppress the procedure code display
`otherfac=`*variable-list*	other factors
`otherint=`*terms*	multi-step interaction terms
`out=`*SAS-data-set*	output experimental design
`procopts=`*options*	PROC OPTEX statement options
`run=`*procedure-list*	list of procedures that can be run
`seed=`*n*	random number seed
`size=`*n* | MIN	candidate-set size
`step=`*n*	step number
`where=`*where-clause*	`where` clause

```
%mktdes(help) /* show option list with ? or help */


%mktdes(factors=x1-x2=2 x3-x5=3, n=18)


%mktdes(factors=x1-x3=2 x4-x6=3 x7-x9=5, n=30)


%mktdes(factors=x1-x3=2, n=30, run=factex,       step=1)
%mktdes(factors=x4-x6=3, n=30, run=factex,       step=2)
%mktdes(factors=x7-x9=5, n=30, run=factex optex, step=3)


%mktdes(factors=x1-x3=2 x4-x6=3(3) x7-x9=4, n=32, size=1024,
        interact=x1|x2|x3@2 x4*x7)
```

MktDups Macro Options

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

```
%mktdups(help) /* show option list with ? or help */


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

%mktdups(linear, factors=x1-x5);


%mktex(2 ** 5, n=2**5)

%mktlab(int=f1-f4, values=1 2)

%choiceff(data=final, model=class(x1-x5), seed=109,
          nsets=42, flags=f1-f4, beta=zero);

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


%mktex(3 ** 9, n=27)

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, model=class(brand x1-x3), seed=420,
          nsets=18, nalts=3, beta=zero);
%mktdups(branded, data=best, factors=brand x1-x3, nalts=3, out=out)
```

MktEval Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`blocks=`*variable*	blocking variable
`data=`*SAS-data-set*	input data set with design
`factors=`*variable-list*	factors in the design
`format=`*format*	format for canonical correlations
`freqs=`*frequency-list*	frequencies to display
`list=`*n*	minimum canonical correlation to list
`outcb=`*SAS-data-set*	within-block canonical correlations
`outcorr=`*SAS-data-set*	canonical correlation matrix
`outfreq=`*SAS-data-set*	frequencies
`outfsum=`*SAS-data-set*	frequency summaries
`outlist=`*SAS-data-set*	list of largest canonical correlations
`print=`*print-options*	controls the display of the results
`vars=`*variable-list*	factors in the design

```
%mkteval(help) /* show option list with ? or help */


%mktex(2 ** 2 3 ** 3, n=18, seed=205)

%mkteval;
```

MktEx Macro Options

Option	Description
`list`	(positional) list of the numbers of factor levels
	(positional) "help" or "?" displays syntax summary
`anneal=`*n1 < n2 < n3 >>*	starting probability for annealing
`annealfun=`*function*	annealing probability function
`anniter=`*n1 < n2 < n3 >>*	first annealing iteration
`balance=`*n*	maximum level-frequency range
`big=`*n < choose >*	size of big full-factorial design
`canditer=`*n1 < n2 >*	iterations for OPTEX designs
`cat=`*SAS-data-set*	input design catalog
`detfuzz=`*n*	determinants change increment
`examine=`< I > < V > <*aliasing* >	matrices that you want to examine
`exchange=`*n*	number of factors to exchange
`fixed=`*variable*	indicates runs that are fixed
`holdouts=`*n*	adds holdout observations
`imlopts=`*options*	IML PROC statement options
`init=`*SAS-data-set*	initial input experimental design
`interact=`*interaction-list*	interaction terms
`iter=`*n1 < n2 < n3 >>*	maximum number of iterations
`levels=`*value*	assigning final factor levels
`maxdesigns=`*n*	maximum number of designs to make
`maxiter=`*n1 < n2 < n3 >>*	maximum number of iterations
`maxstages=`*n*	maximum number of algorithm stages
`maxtime=`*n1 < n2 < n3 >>*	approximate maximum run time
`mintry=`*n*	minimum number of rows to process
`mutate=`*n1 < n2 < n3 >>*	mutation probability
`mutiter=`*n1 < n2 < n3 >>*	first iteration to consider mutating
`n=`*n*	number of runs in the design
`options=accept`	accept designs that violate restrictions
`options=check`	checks the efficiency of the `init=` design
`options=file`	renders the design to a file
`options=int`	add an intercept variable `x0` to the design
`options=justinit`	stop processing after making the initial design
`options=largedesign`	stop after `maxtime=` minutes have elapsed
`options=lineage`	displays the lineage of the orthogonal array
`options=nodups`	eliminates duplicate runs
`options=nofinal`	skips displaying the final efficiency
`options=nohistory`	does not display the iteration history
`options=nooadups`	check for orthogonal array with duplicate runs
`options=noqc`	do not use the SAS/QC product
`options=nosort`	does not sort the design
`options=nox`	suppress the `x`*n* scalars with restrictions
`options=quick`	`optiter=0`, `maxdesigns=2`, `unbalanced=0`, `tabiter=1`
`options=quickr`	`optiter=0`, `maxdesigns=1`, `unbalanced=0`, `tabiter=0`
`options=quickt`	`optiter=0`, `maxdesigns=1`, `unbalanced=0`, `tabiter=1`
`options=render`	displays the design compactly in the SAS listing

Option	Description
`options=refine`	with `init=`, never reinitializes
`options=resrep`	reports on the progress of the restrictions
`options=+-`	renders –1 as – and 1 as + in two-level factors
`options=3`	applies `options=+-` to 3-level factors
`options=512`	adds some designs in 512 runs
`optiter=`*n1 < n2 >*	OPTEX iterations
`order=`*value*	coordinate exchange column order
`out=`*SAS-data-set*	output experimental design
`outall=`*SAS-data-set*	output data set with all designs
`outeff=`*SAS-data-set*	output data set with final efficiency
`outr=`*SAS-data-set*	randomized output experimental design
`partial=`*n*	partial-profile design
`repeat=`*n1 n2 n3*	times to iterate on a row
`reslist=`*list*	constant matrix list
`resmac=`*macro-name*	constant matrix creation macro
`restrictions=`*macro-name*	restrictions macro
`ridge=`*n*	ridging factor
`seed=`*n*	random number seed
`stopearly=`*n*	the macro can stop early
`tabiter=`*n1 < n2 >*	design table initialization iterations
`tabsize=`*n*	orthogonal array size
`target=`*n*	target efficiency criterion
`unbalanced=`*n1 < n2 >*	unbalance initial design iterations

```
%mktex(help) /* show option list with ? or help */

%mktex(2 3 ** 7, n=18)

%mktex(n=104, options=int)

%mktex(2 ** 5  3 ** 4  5 5 5  6 6, n=60, seed=104, maxtime=.3 .3 .3)

%macro resmac;
   client    = {1.29 1.69 2.09 .}[x1];
   extension = {1.39 1.89 2.39 .}[x2];
   regional  = {1.99 2.49      .}[x5];
   private   = {1.49 2.29      .}[x6];
   national  = {1.99 2.39      .}[x8];
   navail    = (client  ^= .) + (extension ^= .) + (regional ^= .) +
               (private ^= .) + (national  ^= .);
   if (navail < 2) | (navail > 4) then bad = abs(navail - 3);
   else                                bad = 0;
   %mend;

%mktex( 4 4 2 2 3 3 2 3, n=26, interact=x2*x3 x2*x4 x3*x4 x6*x7,
        restrictions=resmac, seed=377 )
```

MktKey Macro Options

Option	Description
`list`	(positional) variable list or n rows and n columns
	(positional) "help" or "?" displays syntax summary

```
%mktkey(help) /* show option list with ? or help */

%mktkey(3 5)

%mktkey(5 10 t)

%mktkey(x1-x20)
```

MktLab Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`cfill=`*character-string*	character fill value
`data=`*SAS-data-set*	input design data set
`dolist=`*do-list*	new values using a do-list syntax
`int=`*variable-list*	name of an intercept variable
`key=`*SAS-data-set*	key data set
`labels=`*macro-name*	macro that provides labels and formats
`nfill=`*number*	numeric fill value
`options=noprint`	suppress the display of the variable mappings
`out=`*SAS-data-set*	output data set with recoded design
`prefix=`*variable-prefix*	prefix for naming variables
`statements=`*SAS-code*	add extra statements
`values=`*value-list*	the new values for all of the variables
`vars=`*variable-list*	list of variable names

```
%mktlab(help) /* show option list with ? or help */


%mktex(n=12, options=nosort)

%mktlab(data=design, values=-1 1, int=Had0, prefix=Had);


%mktex(n=18, seed=17)

%mktblock(nblocks=2, factors=x1-x4)

data key;
   input Brand $ Price Size;
   format price dollar5.2;
   datalines;
Acme 1.49   6
Apex 1.79   8
.    1.99  12
;

%mktlab(data=blocked, key=key)
```

MktMDiff Macro Options

Option	Description
`layout`	(positional) choice data set structure
	(positional) "help" or "?" displays syntax summary
`attrs=`*macro-variable*	macro variable with attribute descriptions
`b=`*b*	number of sets (alias for `nsets=`)
`classopts=`*options-list*	class variable options
`data=`*SAS-data-set*	input data set
`design=`*SAS-data-set*	design data set
`group=`*k*	number of groups in the `design=` data set
`k=`*k*	set size (alias for `setsize=`)
`nattrs=`*t*	number of attributes (alias for `t=`)
`nsets=`*b*	number of sets (alias for `b=`)
`options=nocode`	just create the OUT= data set
`options=noanalysis`	just create the OUT= and OUTCODED= data sets
`options=nosort`	do not sort the parameter estimates table
`options=nodrop`	do not drop the variable or parameter column
`out=`*SAS-data-set*	merged data and design data set
`outcoded=`*SAS-data-set*	coded data set
`outparm=`*SAS-data-set*	parameter estimates data set
`rescale=`*value*	table of rescaled parameter estimates
`setsize=`*k*	set size (alias for `k=`)
`t=`*t*	number of attributes or messages (alias for `nattrs=`)
`vars=`*variable-list*	`data=` data set variables

```
%mktmdiff(help) /* show option list with ? or help */

%mktbibd(t=9, k=5, b=18, seed=377, out=sasuser.bibd)

title "Best Worst Example with Cell Phone Attributes";
data bestworst;
   input Sub $ @4 (b1-b18 w1-w18) (1.);
   datalines;
 1 188661884399349653941955342212935494
 2 765358873891388493922673644336595554
 3 782126282892848564995993447213935655
 4 481363264246399187162125415351281453
 5 787168863811878175225995382352235293
 6 787658867891878667495965442313345453
 7 788171867736888187465395344393344453
 8 788771867711888687445353445353335254
 9 188778887896878687425323443242344454
10 788778887816888687445353444343344454
;
```

```
%let attributes=Camera,Flip,Hands Free,Games,Internet
,Free Replacement,Battery Life,Large Letters,Applications;
%phchoice( on )

%mktmdiff(bw, b=18, t=9, k=5, attrs=attributes,
          data=bestworst, design=sasuser.bibd)
```

MktMerge Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`blocks=`1 | *variable*	blocking variable
`data=`*SAS-data-set*	input SAS data set
`design=`*SAS-data-set*	input SAS choice design data set
`nalts=`*n*	number of alternatives
`nsets=`*n*	number of choice sets
`out=`*SAS-data-set*	output SAS data set
`setvars=`*variable-list*	variables with the data
`statements=`*SAS-statements*	additional statements

```
%mktmerge(help) /* show option list with ? or help */

%mktex(3 ** 4, n=18, seed=104)

data results;
   input Subj (choose1-choose18) (1.) @@;
   datalines;
  1 333542334333314443  2 333212344333333345  3 333212333333313333
  4 133242144334414453  5 335242134333513443  6 333242234333314443
  7 333432334332323443  8 333242234334414443  9 333432331352313343
;

data key;
   input Brand $ & Price $;
   datalines;
Brand 1  x1
Brand 2  x2
Brand 3  x3
Brand 4  x4
Another  .
;

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

%mktmerge(design=rolled, data=results, out=res2,
          nsets=18, nalts=5, setvars=choose1-choose18)
```

MktOrth Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`filter=`*n*	extra filtering of the design catalog
`maxn=`*n*	maximum number of runs of interest
`maxlev=`*n*	maximum number of levels
`options=lineage`	construct the design lineage
`options=mktex`	the macro is being called from the `%MktEx` macro
`options=mktruns`	the macro is being called from the `%MktRuns` macro
`options=parent`	lists only parent designs
`options=dups`	suppress duplicate and inferior design filtering
`options=512`	adds some designs in 512 runs
`outall=`*SAS-data-set*	output data set with all designs
`outcat=`*SAS-data-set*	design catalog data set
`outlev=`*SAS-data-set*	output data set with the list of levels
`range=`*range-specification*	number of runs of interest

```
%mktorth(help) /* show option list with ? or help */


%mktorth(range=n eq 12 or n eq 18);

proc print; run;


%mktorth(range=n=72)

proc print data=mktdeslev(where=(x2 ge 5 and x3 ge 5 and x4 and x6));
   var n design reference;
   run;


%mktorth(maxn=100, maxlev=6)


%mktorth(range=n=36, options=lineage)
proc print; run;


%mktorth(range=n=36, options=parent)
proc print; run;
```

MktPPro Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`ibd=`*SAS-data-set*	incomplete block design
`design=`*SAS-data-set*	orthogonal array
`out=`*SAS-data-set*	output partial-profile design
`print=`*print-options*	output display options
`x=`*SAS-data-set*	incomplete block design incidence matrix

```
%mktppro(help) /* show option list with ? or help */

%mktex(2 4 2 2 2, n=8)

data design; set design; drop x2; run;

%mktbibd(b=20, nattrs=16, setsize=4, seed=104, out=b)

%mktppro(ibd=b, print=f p)

%choiceff(data=chdes, model=class(x1-x16 / sta), nsets=80, nalts=2,
          options=relative, beta=zero, init=chdes, initvars=x1-x16)
```

MktRoll Macro Options

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

```
%mktroll(help) /* show option list with ? or help */

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

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

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

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

MktRuns Macro Options

Option	Description
`list`	(positional) numbers of levels of all the factors
	(positional) "help" or "?" displays syntax summary
`interact=`*interaction-list*	interaction terms
`max=`n $< m >$ $<$ f $>$	largest design sizes to try
`n=`n	design size to evaluate
`maxlev=`n	maximum number of levels
`maxoa=`n	maximum number of orthogonal arrays
`options=justparse`	used by other `Mkt` macros to parse the list
`options=multiple`	allow terms to be counted multiple times
`options=multiple2`	`option=multiple` and more detailed output
`options=noprint`	suppress the display of all output
`options=nosat`	suppress the saturated design from the design list
`options=source`	displays source of numbers in design sizes
`options=512`	adds some designs in 512 runs
`out=`*SAS-data-set*	data set with the suggested sizes
`outorth=`*SAS-data-set*	data set with orthogonal array list
`toobig=`n	specifies problem that is too big

```
%mktruns(help) /* show option list with ? or help */

%mktruns(2 2 2 3 3 3 3)

%mktruns(2**10 3**3, interact=11|12|13@2)
```

Paint Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`values=`*do-list*	input data values
`var=`*variable-name*	input variable
`data=`*SAS-data-set*	input SAS data set
`out=`*SAS-data-set*	output SAS data set
`macro=`*macro-name*	SAS macro name
`colors=`<*color-list*> <*data-value-list*>	colors list
`level=`*measurement-level*	measurement level of the data
`symbols=`*symbols-list*	list of symbols
`format=`*SAS-format*	nominal and ordinal variable format
`order=`*summary-order*	nominal and ordinal variable order
`select=`*n*	selection state
`show=`*n*	show/hide state
`include=`*n*	include/exclude state
`label=`*n*	label/unlabel state
`rgbround=`*rounding-list*	RGB value rounding instructions
`missing=`*missing-specification*	missing value handling
`debug=`*debug-string*	debugging information to display

```
%paint(help) /* show option list with ? or help */

data cars(drop=i);
   do i = 1 to 7;
      input (MPG Reliability Acceleration Braking Handling Ride
             Visibility Comfort Quiet Cargo) (1.) +1 @;
      output;
      end;
   datalines;
3334444544 2434453555 2354353545 3244443424 2354553543 3232423224 4145555422
2254541241 2445353555 5335425223 4155555525 2153542242 3333444544 3253353544
;

%paint(values=-1 to 1 by 0.05, macro=setstyle,
       colors=CXEEEEEE red magenta blue cyan white cyan blue magenta red CXEEEEEE
              -1 -0.99 -0.75 -0.5 -0.25 0 0.25 0.5 0.75 0.99 1)

proc template;
   edit Base.Corr.StackedMatrix;
      column (RowName RowLabel) (Matrix) * (Matrix2);
      edit matrix;    %setstyle(backgroundcolor)    end;
      end;
   run;

ods html body="corr.html" style=statistical;
proc corr data=cars noprob;    run;
ods html close;
proc template;   delete Base.Corr.StackedMatrix;   run;
```

PHChoice Macro Options

Option	Description
`onoff`	(positional) `on`, `off`, or `expb`
	(positional) "help" or "?" displays syntax summary
`column`	(positional) list of columns

```
%phchoice(help) /* show option list with ? or help */

%phchoice(on)

%phchoice(off)

%phchoice(expb)
```

PlotIt Macro Options

Option	Description
`help`	(positional) "help" or "?" displays syntax summary
`adjust1=`*SAS-statements*	adjust the preprocessing data set
`adjust2=`*SAS-statements*	includes statements with PROC PLOT
`adjust3=`*SAS-statements*	extra statements for the final DATA step
`adjust4=`*SAS-statements*	extra statements for the final DATA step
`adjust5=`*SAS-statements*	extra statements for the final DATA step
`antiidea=`*n*	eliminates PREFMAP anti-ideal points
`blue=`*expression*	blue part of RGB colors
`bright=`*n*	generates random label colors
`britypes=`*type*	types to which `bright=` applies
`cframe=`*color*	color of background within the frame
`cirsegs=`*n*	circle smoothness parameter
`color=`*color*	default color
`colors=`*colors-list*	default label and symbol color list
`cursegs=`*n*	number of segments in a curve
`curvecol=`*color*	color of curve
`data=`*SAS-data-set*	input data set
`datatype=`*data-type*	data analysis that generated data set
`debug=`*values*	debugging output
`excolors=`*color-list*	excludes from the Annotate data set
`extend=`*axis-extensions*	extend the x and y axes
`extraobs=`*SAS-data-set*	extra observations data set
`exttypes=`*type*	types for `extraobs=` data set
`filepref=`*prefix*	file name prefix
`font=`*font*	default font
`framecol=`*color*	color of frame
`gdesc=`*description*	catalog description
`gname=`*name*	catalog entry
`gopplot=`*goptions*	`goptions` for plotting to screen
`gopprint=`*goptions*	`goptions` for printing
`gopts2=`*goptions*	`goptions` that are always used
`gopts=`*goptions*	additional `goptions`
`gout=`*catalog*	`proc ganno gout=` catalog
`green=`*expression*	green part of RGB colors
`hminor=`*n* | *do-list*	horizontal axis minor tick marks
`hnobs=`*n*	horizontal observations for contour plots
`hpos=`*n*	horizontal positions in graphics area
`href=`*do-list*	horizontal reference lines
`hsize=`*n*	horizontal graphics area size
`inc=`*n*	`haxis=by inc`, `vaxis=by inc`
`interpol=`*method*	axis interpolation method
`labcol=`*label-colors*	colors for the point labels
`label=`*label-statement*	`label` statement
`labelcol=`*color*	color of variable labels
`labelvar=`*label-variable*	point label variable

Option	Description
`labfont=`*label-fonts*	fonts for the point labels
`labsize=`*label-sizes*	sizes for the point labels
`ls=`*n*	how line sizes are generated
`lsinc=`*n*	increment to line size
`lsizes=`*number-list*	line sizes (thicknesses)
`makefit=`*n*	proportion of graphics window to use
`maxiter=`*n*	maximum number of iterations
`maxokpen=`*n*	maximum acceptable penalty sum
`method=`*value*	where to send the plot
`monochro=`*color*	overrides all other colors
`nknots=`*n*	number of knots option
`offset=`*n*	move symbols for coincident points
`options=border`	draws a border box
`options=close`	close up the border box and the axes
`options=diag`	draws a diagonal reference line
`options=expand`	expands the plot to fill the window
`options=noback`	do not set the frame color
`options=nocenter`	do not center
`options=noclip`	do not clip
`options=nocode`	suppress the PROC PLOT and `goptions` statements
`options=nodelete`	do not delete intermediate data sets
`options=nohistory`	suppress the iteration history table
`options=nolegend`	suppress the display of the legends
`options=noprint`	`nolegend`, `nocode`, and `nohistory`
`options=square`	the same ticks for both axes and a square plot
`options=textline`	lines overwrite text
`out=`*SAS-data-set*	output Annotate data set
`outward=none` | *char*	PLOT statement `outward=`
`paint=`*interpolation*	color interpolation
`place=`*placement*	generates a `placement=` option
`plotopts=`*options*	PLOT statement options
`plotvars=`*variable-list*	*y*-axis and *x*-axis variables
`post=`*filename*	graphics stream file name
`preproc=`*SAS-data-set*	preprocessed `data=` data set
`procopts=`*options*	PROC PLOT statement options
`ps=`*n*	page size
`radii=`*do-list*	radii of circles
`red=`*expression*	red part of RGB colors
`regdat=`*SAS-data-set*	intermediate regression results data set
`regopts=`*options*	regression curve fitting options
`regprint=`*regression-option*	regression options
`rgbround=`*RGB-rounding*	`paint=` rounding factors
`rgbtypes=`*type*	types to which `paint=` and RGB options apply
`style=`A | B	obsolete option not supported
`symbols=`*symbol-list*	plotting symbols
`symcol=`*symbol-colors*	colors of the symbols
`symfont=`*symbol fonts*	symbol fonts
`symlen=`*n*	length of the symbols

Option	Description
`symsize=`*symbol-sizes*	sizes of symbols
`symtype=`*symbol-types*	types of symbols
`symvar=`*symbol-variable*	plotting symbol variable
`tempdat1=`*SAS-data-set*	intermediate results data set
`tempdat2=`*SAS-data-set*	intermediate results data set
`tempdat3=`*SAS-data-set*	intermediate results data set
`tempdat4=`*SAS-data-set*	intermediate results data set
`tempdat5=`*SAS-data-set*	intermediate results data set
`tempdat6=`*SAS-data-set*	intermediate results data set
`tickaxes=`*axis-string*	axes to draw tick marks
`tickcol=`*color*	color of ticks
`tickfor=`*format*	tick format used by `interpol=tick`
`ticklen=`*n*	length of tick mark in horizontal cells
`titlecol=`*color*	color of title
`tsize=`*n*	default text size
`types=`*observation-types*	observations types
`typevar=`*variable*	observation types variable
`unit=in` | `cm`	`hsize=` and `vsize=` unit
`vechead=`*vector-head-size*	how to draw vector heads
`vminor=`*n* | *do-list*	vertical axis minor tick marks
`vnobs=`*n*	vertical observations for contour plots
`vpos=`*n*	vertical positions in graphics area
`vref=`*do-list*	vertical reference lines
`vsize=`*n*	vertical graphics area size
`vtoh=`*n*	PROC PLOT `vtoh=` option
`xmax=`*n*	maximum horizontal graphics area size
`ymax=`*n*	maximum vertical graphics area size

```
%plotit(help) /* show option list with ? or help */

data smoke;
   length Employee $ 20;
   input None Light Medium Heavy @13 Employee $ & Weight No Yes;
   label none    = "Nonsmoker"       light   = "Light Smoker"
         medium  = "Medium Smoker"   heavy   = "Heavy Smoker"
         no      = "Nondrinker"      yes     = "Drinker";
   datalines;
 4  2  3  2 Senior Manager   1  0 11
 4  3  7  4 Junior Manager   1  1 17
25 10 12  4 Senior Employee  1  5 46
18 24 33 13 Junior Employee  1 10 78
10  6  7  2 Admin            1  7 18
42 29 20  9 Nation          -1  .  .
;
```

```
proc corresp outc=c;
   var    none -- heavy no yes;
   id     employee;
   weight weight;
   supvar no yes;
   run;

%plotit(data=c, datatype=corresp)
```