desmat — Interactions and contrasts

John Hendrickx, Management Studies Group, Wageningen University, The Netherlands

Last update: November 11, 2001

Syntax

`desmat` *model* `[, colinf defcon(`*contrast_specification*`) ]`

`desmat:` *stata_procedure depvar model* `[using]` *[if] [in] [fweight pweight aweight iweight]*
`[, verbose defcon(`*contrast_specification*`) desrep(`*desrep_options*`)`
procedure_options `]`

Description

`desmat` is a replacement for `xi`, allowing higher order interactions and different types of contrasts for categorical variables. `desmat` generates a set of dummy variables `_x_*` for the model being specified. A companion program `desrep` is used to present the results with descriptive labels. A second program `destest` is used to perform Wald tests on model terms. A third program `showtrms` lists the dummy variables created by `desmat`, together with the terms these pertain to and the types of contrasts used.

Like `xi`, `desmat` can be used as either as a command or as a command prefix. When used as a command, `desmat` generates a set of dummy variables for use by subsequent Stata programs. When used as a command prefix, the model is estimated and presented using `desrep`, with no standard Stata output.

The model

The *model* consists of one or more *terms* separated by spaces. A *term* can be a single variable, two or more variables joined by periods, or two or more variables joined by asterisks. A period is used to specify an interaction effect as such, whereas an asterisk indicates hierarchical notation, in which both the interaction effect itself plus all possible nested interactions and main effects are included. For example, the term "`vote*educ*race`" is expanded to "`vote educ vote.educ race vote.race educ.race vote.educ.race`".

Variables may be either string or numeric. All variables in the model are treated as categorical unless specified otherwise (see the section on "Contrasts" below). A variable may be prefixed by an "`@`" to flag it as a continuous variable. For example:

```
desmat: regress brate @medage @medagesq region
```

The variables `medage` and `medage` will be treated as continuous variables. The variable region will be treated as categorical and dummy variables will be generated using its first category as reference category.

Options

When desmat is used as a command prefix to a Stata procedure, if or in options as well as weights may be specified in the usual manner and will be passed on to the procedure. Any options besides verbose, defcon and desrep will be passed on as well.

using *filename*
: If "using *filename*" is specified then the results will be written to a tab-delimited ascii file. The default extension for filename is ".out" (cf. outsheet). See desrep below for further details. Place "using *filename*" after the model specification.

defcon(*contrast specification*)
: By default, desmat generates dummy variables using the first category as reference category. The defcon option can be used to specify a different contrast using the contrast specifications discussed below.

Options for desmat as a command prefix

verbose
: When used as a command prefix, desmat produces no output, estimates the model quietly, then calls desrep to display the results. The verbose option can be used to print intermediate results.

desrep(*desrep options*)
: The desrep option can be used to pass options on to desrep after the model has been estimated but prior to the presentation of results. Note that most of these options can be specified using global macro variables; see the section below on desrep for details. An exception could be the exp option. desrep displays linear coefficients even if the procedure prints exponential coefficients, e.g. the odds-ratios produced by logistic. Specify:

```
desmat: logistic vote memb educ*race [fw=pop], desrep(exp all)
```

to display odds-ratios. See the section on desrep for further details.

Options for desmat as as command by itself

For compatibility with earlier versions, a default contrast may be specified as an option rather than an argument for the defcon option when desmat is used as a command by itself.

colinf
: When interaction terms are specified, desmat will often generate duplicate dummy variables. These duplicates are subsequently removed by dropping collinear variables. In some cases however, it can occur that variables are unexpectedly dropped. The conlinf option can be used to produce a report on which variables have been removed.

Contrasts

By default, desmat generates dummy variables using the first category as the reference category, as does xi. However, it can also use different types of restrictions (contrasts) and different reference categories when generating the dummy variables. A restriction of some

type is required for the effects of categorical variables to be identifiable. The restriction used does not affect the fit of the model but does determine the meaning of the parameters. A common restriction and the one used by `xi` is to drop the dummy variable for a reference category. The parameters for that variable are then relative to the reference category. Another common constraint is the deviation contrast, in which parameters have a sum of zero. One parameter can therefore be dropped as redundant during estimation and found afterwards using minus the sum of the estimated parameters, or by re-estimating the model using a different omitted category. Bock (1975) and Finn (1974) discuss other types of parameterizations (or contrasts) and the technical details in implementing them.

A parameterization can be specified as a name, of which the first three characters are significant, optionally followed by a specification of the reference category in parentheses (no spaces). The reference category should refer to the category *number*, not the category *value*. So for a variable with values 0 to 3, the parameterization "dev(1)" indicates that the deviation contrast is to be used with the first category (i.e. 0) as the reference. If no reference category is specified, or the reference category is less than 1 then the first category is used as reference category. If value specified is larger than the number of categories then the highest category is used. Note that for certain types of parameterizations, the "reference" specification has a different meaning.

The available parameterization types are:

`ind(`*ref*`)` Indicator contrast, i.e. dummy variables with *ref* as reference (omitted) category. This is the parameterization used by `xi` and is the default parameterization for `desmat`.

`dir` A direct effect, i.e. used to include continuous variables in the model.

`dev(`*ref*`)` Deviation contrast. Parameters sum to zero over the categories of the variable. The parameter for *ref* is omitted as redundant, but can be found from minus the sum of the estimated parameters.

`sim(`*ref*`)` Simple contrast with *ref* as reference category. The highest order effects are the same as indicator contrast effects, but lower order effects and the constant will be different.

`dif(`*ref*`)` Difference contrast, for ordered categories. Parameters are relative to the next category. If the first letter of *ref* is "b" then the backward difference contrast is used instead, and parameters are relative to the previous category.

`hel(`*ref*`)` Helmert contrast, for ordered categories. Estimates represent the contrast between that category and the mean value for the remaining categories. If the first letter of *ref* is "b" then the reverse Helmert contrast is used instead, and parameters are relative to the mean value of the preceding categories.

`orp(`*ref*`)` Orthogonal polynomials of degree *ref*. The first category is a linear effect, the second quadratic, etc. This option calls `orthpoly` to generate the design (sub)matrix.

`use(`*ref*`)` A user-defined contrast. *Ref* refers to a contrast matrix with the same number of columns as the variable has categories, and at least one less rows. If `rownames`

are specified for this matrix, these names will be used as variable labels for the resulting dummy variables. [Single lowercase letters as names for the contrast matrix cause problems at the moment, e.g. "`use(c)`". Use uppercase names or more than one letter, e.g. "`use(cc)`" or "`use(C)`"]

Specifying contrasts using the `defcon` option

The defcon option can be used to specify a different contrast than "`ind(1)`" for all variables in all terms, e.g.

```
desmat: logistic vote memb educ*race [fw=pop], desrep(exp all)
defcon(dev(99))
```

The deviation contrast will now be used with the highest category as the redundant category.

The global variable `$D_CON` can be used to specify a default contrast for the current Stata session. For example:

```
global D_CON "dev(99)"
```

will cause `desmat` to use the deviation contrast for the duration of the Stata session. By specifying this command in their `profile.do`, users can specify a different contrast for all `desmat` models. The `$D_CON` global variable is overridden by the `defcon` option if this is specified.

Specifying contrasts using the `pzat` characteristic

A `pzat` characteristic can be assigned to a variable to specify a contrast to be used for that variable. For example, to use the backward difference contrast for education but the default indicator contrast for the other variables, use:

```
char educ[pzat] dif(b)
desmat: logistic vote memb educ*race [fw=pop], desrep(exp all)
```

The `pzat` characteristic will override the contrast specified by the `defcon` option. So in

```
char educ[pzat] dif(b)
desmat: logistic vote memb educ*race [fw=pop], desrep(exp all)
defcon(dev(99))
```

The difference contrast will be used for all variables except `educ`.

Specifying contrasts in the model statement

It is also possible to specify contrasts in the model statement, on a variable by variable basis if so desired. This is done by appending "`=`*con*`[(`*ref*`)]`" to a single variable, "`=`*con*`[(`*ref*`)].`*con*`[(`*ref*`)]`" to an interaction effect, and "`=`*con*`[(`*ref*`)]*`*con*`[(`*ref*`)]`" to an interaction using hierarchical notation. A somewhat silly example:

```
desmat race=ind(1) educ=hel memb vote vote.memb=dif.dev(1), defcon(ind(99))
```

The indicator contrast with the highest category as reference will be used for "`memb`" and "`vote`". The variable "`race`" will use the indicator contrast as well but with the first category as reference, other effects will use the contrasts specified. Interpreting this mishmash of parameterizations would be quite a chore of course.

A variable's `pzat` characteristic overrides the `defcon` option, but is itself overridden by a specification in the model. For example:

```
char educ[pzat] dif(b)
desmat vote*memb vote*educ*race=dev(99)*orp(1)*dev(99) educ*race*memb,
defcon(dev(99))
```

`Educ` will use a first-degree polynomial restriction in the `vote*educ*race` term and a backward difference contrast elsewhere. All other variables will use the deviation contrast.

Specifying contrasts in the model statement will tend to look messy and provides overkill in flexibility. Use of the `pzat` characteristic in conjunction with the defcon option and the `@` prefix to flag continuous variables will usually be preferable.

The simple contrast versus the indicator contrast

The simple contrast and the indicator contrast both use a reference category. What then is the difference between the two? Both produce the same parameters and standard errors for the highest order effect. In the example below, the estimates for `vote.educ.race` and `educ.race.memb` are the same whether the simple or indicator contrast is used, but all other estimates are different.

The difference is that the parameters for the indicator contrast are relative to the reference category whereas the values for the simple contrast are actually relative to the mean value within the categories of the variable. For example, the systolic blood pressure (`systolic` in `systolic.dta`; see e.g. ANOVA) has the following mean values for each of the four categories of the variable `drug`: 26.06667, 25.53333, 8.75, and 13.5. In a oneway analysis of `systolic` by `drug`, the constant using the indicator contrast with the first category as reference is 26.067. Using the simple contrast, the constant is 18.4625, the mean of the four category means, regardless of the reference category.

Calculating predicted values is a good deal more involved using the simple contrast. Using the simple contrast, `drug` has the following codings:

	b1	b2	b3
drug==1	-.25	-.25	-.25
drug==2	.75	-.25	-.25
drug==3	-.25	.75	-.25
drug==4	-.25	-.25	.75

Given the estimates `_cons` = 18.463, `b1` = -.533, `b2` = -17.317, `b3` = -12.567, the predicted values are calculated as:

```
drug==1:   18.463  -.250*-.533   -.250*-17.317   -.250*-12.567  =   26.067
drug==2:   18.463   .750*-.533   -.250*-17.317   -.250*-12.567  =   25.533
drug==3:   18.463  -.250*-.533  +.750*-17.317   -.250*-12.567  =    8.750
drug==4:   18.463  -.250*-.533   -.250*-17.317  +.750*-12.567  =   13.500
```

If the indicator contrast is used, the b-parameters have the same value but the constant is 26.067, the mean for category 1. The predicted values can be calculated simply as

```
drug==1:   26.067                                              =   26.067
drug==2:   26.067       -.533                                  =   25.533
drug==3:   26.067                    -17.317                  =    8.750
drug==4:   26.067                                  -12.567    =   13.500
```

The flip side of this is that lower order effects will depend on the choice of reference category if the indicator contrast is used but not for the simple contrast. Tests for the significance of lower order terms will also depend on the reference category if the indicator contrast is used.

In the sample program below, `destest` will produce different results for all terms except the highest order, `vote.educ.race` and `educ.race.memb`, if another reference category is used. Using the simple contrast, or one of the other pre-defined contrasts such as the deviation or difference contrast, will give the same results for these tests.

Example

Knoke & Burke (1980: 23) present a four-way table of race by education by membership by vote turnout. Their loglinear model {VM}{VER}{ERM} could be specified as:

```
desmat: glm pop vote*memb vote*educ*race educ*race*memb, link(log)
family(poisson)
```

`desmat` produces the following output:

```
------------------------------------------------------------------------------
   Generalized Linear Models
------------------------------------------------------------------------------
   Dependent variable                                                    pop
   Variance function:                                                Poisson
   Link function:                                                        Log
   Optimization:                                      ML: Newton-Raphson
   Number of observations:                                              24
   Deviance:                                                         4.756
   Deviance dispersion:                                              0.951
   Log likelihood:                                                 -65.841
   Model degrees of freedom:                                            18
   Residual degrees of freedom:                                          5
   AIC:                                                              7.070
   BIC:                                                            -55.627
   Prob:                                                             0.446
------------------------------------------------------------------------------
nr Effect                                                  Coeff        s.e.
------------------------------------------------------------------------------
   pop
     vote
1      Voted                                              -0.021       0.111
     memb
2      One or More                                        -0.536**     0.119
     vote.memb
3      Voted.One or More                                   0.768**     0.120
     educ
4      High School Graduate                               -0.488**     0.126
5      College                                            -1.638**     0.170
     vote.educ
6      Voted.High School Graduate                          0.192       0.141
7      Voted.College                                       0.844**     0.164
     race
8      Black                                              -1.441**     0.191
     vote.race
9      Voted.Black                                        -0.070       0.244
     educ.race
10     High School Graduate.Black                         -0.946*      0.384
11     College.Black                                      -0.465       0.478
     vote.educ.race
12     Voted.High School Graduate.Black                    0.500       0.432
13     Voted.College.Black                                -0.723       0.432
     educ.memb
14     High School Graduate.One or More                    0.648**     0.138
15     College.One or More                                 1.397**     0.161
     race.memb
16     Black.One or More                                  -0.525*      0.253
     educ.race.memb
```

```
17     High School Graduate.Black.One or More                  0.170       0.410
18     College.Black.One or More                               0.783       0.507
19   _cons                                                     4.781**     0.085
------------------------------------------------------------------------------
*  p < .05
** p < .01
```

Alternatively, `desmat` can be used as a command by itself to generate a set of dummy variables, for use in a subsequent Stata procedure.

```
desmat vote*memb vote*educ*race educ*race*memb
glm pop _x_*, link(log) family(poisson)
desrep
```

This produces the same output, supplemented by a report by `desmat` on the dummy variables it had generated and the output generated by `glm` itself. This effect can also be obtained by specifying the `verbose` when using `desmat` in command prefix mode.

The program `destest` can be used after estimating a model to perform a Wald test on selected model terms. In `destest`, using an asterisk in model terms does not cause nested effects to be tested as well. To test these as well, they must be explicitly listed, or `destest` should be used without any arguments to test *all* model terms.

```
. * test only the highest order terms
. destest vote*memb vote*educ*race educ*race*memb
------------------------------------------------------------------------------
Term                                                Wald chi2      df  P > chi2
------------------------------------------------------------------------------
vote.memb                                             41.146**      1     0.000
vote.educ.race                                         5.978        2     0.050
educ.race.memb                                         2.391        2     0.303
------------------------------------------------------------------------------
*  p < .05
** p < .01
```

desrep

`desrep` is a program for viewing the results of Stata estimation commands. It can be used after estimating any model but is particularly useful in conjunction with `desmat`. `desrep` is called by `desmat` when this is used in command prefix mode. In that case, options for formatting the output can be specified using the `desrep` option in `desmat`.

By default, `desrep` prints only the coefficients, their standard errors, and symbols indicating significance, thus allowing longer descriptive labels. Used in conjunction with `desmat`, `desrep` will print labels on model terms and category values using the `[varn]` and `[valn]` characteristics `desmat` assigns to its dummy variables. If `desmat` was not used, variable labels are printed instead. Estimates are preceded by a summary of model information, based on results saved in `e()` by the command.

Syntax for desrep

```
desrep [using filename] [ , fw(#) ndec(#) sigcut(numlist) sigsym(list)
  sigsep(#) nrwd(#) modinfo sig se zval prob ci all notrunc exp outraw
  replace ]
```

By default, desrep prints model information, coefficients, standard errors, and symbols indicating the significance. Additional statistics can be requested and printing of standard errors and significance symbols can be suppressed. Defaults for some of these options can be modified using global macro variables (see below).

If "using *filename*" is specified then the results are written to a tab-delimited ascii file. The default extension for filename is ".out" (cf. outsheet). If filename already exists, desrep will attempt to find a valid filename by appending a number (this is done using the included outshee2.ado program). The replace option can be used to overwrite an existing file.

Options

A number of options can be used to specify which results are printed and how they are formatted.

fw(#)
: "Field Width", used to specify the number of columns used to display the estimates, standard errors, and other requested statistics. Default=10.

ndec(#)
: Specifies the number of decimal places. Default=3.

sigcut(*numlist*)
: desrep places a symbol next to coefficients to indicate whether these are significant at a certain level. The sigcut option is used to specify these levels of significance. The *numlist* should contain a list of values in descending order with the same number of elements as the string list in sigsym. For example, sigcut(.1 .05 .01 .001) together with sigsym(# * ** ***) will use the symbols "#" for p < .1, "*" for p < .05, "**" for p < .01, and "***" for p < .001. The default is (.05 .01)

sigsym(*list*)
: A set of symbols corresponding with the levels of significance given by sigcut. Default=(* **).

sigsep(#)
: The number of spaces between coefficients and symbols indicating significance. Default=0.

nrwd(#)
: The number of columns reserved for numbering the effects. Specifying nrwd(0) can be used to suppress numbering. Default=3.

modinfo
: Use nomodinfo to suppress printing of information on the model and goodness of fit. Default=modinfo.

sig
: Use nosig to suppress printing of symbols for levels of significance. Default=sig.

se
: Use nose to suppress printing of standard errors. Default=se.

`zval`
: Use `zval` to request printing of z-values for models with a chi^2 statistic, t-values for models with an F statistic. Default=`nozval`.

`prob`
: Use `prob` to request printing of p-values. Default=`noprob`.

`ci`
: Use `ci` to request printing of confidence intervals. Default=`noci`.

`all`
: The option `all` can be used to request all standard Stata output, i.e. standard errors, z or t-statistics, probabilities and confidence intervals. Specifying all is thus equivalent to specifying `zval prob ci`.

`notrunc`
: Very long labels are normally cut off and the rightmost section displayed. Use `notrunc` to suppress this and print estimates on a separate line. Default=`trunc`.

`exp`
: If `exp` is specified, `desrep` will report multiplicative parameters, e.g. incident rate ratios in poisson regression, oddsratios in logistic regression. The parameters are transformed into exp(b) and their standard errors into exp(b)*se, where "b" is the linear estimate and "se" its standard error. Note that if `exp` is not specified, `desrep` will produce the linear estimates even if the procedure produces multiplicative versions, since the procedure stores the linear estimates and covariance matrix in "`e(b)`" and "`e(V)`".

  Earlier versions of `desrep` allowed `exp` to be specified as the only argument. This is still allowed if `exp` is the only argument. If other options are specified, `exp` must be specified as an option.

The following two options apply only if "`using`" has been specified to write the data to a tab-delimited ascii file:

`outraw`
: If `outraw` is specified then the results are written with their default formats, e.g. `%9.0g` for floats. In addition, a tab will be inserted between coefficients and significance symbols. Otherwise, the variables are written with a fixed number of decimal places as specified by the `ndec` option (default 3) and significance symbols are appended to coefficients if `sigsep`=0. Default= `nooutraw`.

`replace`
: Overwrite any existing output file. If not specified, `desrep` appends a number to the filename if it already exists. If no valid name has been found after appending 1 to 20, the process stops and the output is not saved. Default= `noreplace`.

Macro variables to control layout

Macro variables can be use to alter the default for certain `desrep` options. The macro variables will still be overridden by options specified at the `desrep` command. The global variables can be specified once at the beginning of the Stata session or in the user's `profile.do` for all sessions. The following global variables may be defined:

```
$D_FW
$D_NDEC
$D_SIGCUT
$D_SIGSYM
$D_SIGSEP
$D_NRWD
$D_SIG
$D_SE
$D_ZVAL
$D_PROB
$D_CI
$D_ALL
$D_TRUNC
$D_RAW
$D_REPL
```

For example, the following can be used to set the column width for estimates to 8, use 2 decimal places, and symbols and cutpoints for levels of significance:

```
global D_NDEC 2
global D_FW 8
global D_SIGCUT ".1 .05 .01 .001"
global D_SIGSYM "# * ** ***"
```

showtrms

`showtrms` produces a legend of the dummy variables produced by `desmat`, the terms these pertain to, and the contrasts used.

Syntax for showtrms

```
showtrms
```

The `showtrms` command has no options. `showtrms` is called automatically when `desmat` is used as a command by itself or when the verbose option is used with `desmat` as a command prefix. `showtrms` can be used at any point after `desmat` has been used to generate a legend for the last design matrix. For example used above, `showtrms` would produce the following output:

```
. showtrms

Desmat generated the following design matrix:

nr   Variables       Term                        Parameterization
     First    Last

 1    _x_1           vote                        ind(0)
 2    _x_2           memb                        ind(1)
 3    _x_3           vote.memb                   ind(0).ind(1)
 4    _x_4    _x_5   educ                        ind(1)
 5    _x_6    _x_7   vote.educ                   ind(0).ind(1)
 6    _x_8           race                        ind(1)
 7    _x_9           vote.race                   ind(0).ind(1)
 8   _x_10   _x_11   educ.race                   ind(1).ind(1)
 9   _x_12   _x_13   vote.educ.race             ind(0).ind(1).ind(1)
10   _x_14   _x_15   educ.memb                  ind(1).ind(1)
```

```
11   _x_16           race.memb                  ind(1).ind(1)
12   _x_17   _x_18   educ.race.memb              ind(1).ind(1).ind(1)
```

destest

`destest` is for use after estimating a model with a design matrix generated by `desmat` to perform a Wald test on model terms.

Syntax for destest

`destest [`*termlist*`] [using` *filename*`] [,joint equal outraw replace ndec(#)`
`sigcut(`*numlist*`) sigsym(`*list*`) sigsep(#) ]`

The *termlist* consists of one or more terms as specified in `desmat`. A *term* can consist of a single variable, or two or more variables separated by either asterisks or periods. If asterisks are used, they will be changed into periods by `destest`, i.e. only the highest order interaction will be tested. Nested terms will be tested only if they are explicitly included. (This syntax makes it easier to copy the model syntax and test the highest order terms, which is what most people will be interested in). If `destest` is specified without any arguments, all terms from the last `desmat` model will be tested.

If "`using` *filename*" is specified then the results are written to a tab-delimited ascii file. The default extension for filename is "`.out`" (cf. `outsheet`). If filename already exists, `destest` will attempt to find a valid filename by appending a number (this is done using the included `outshee2.ado` program). The `replace` option can be used to overwrite an existing file.

`desmat` creates global macro variables "`$term1`", "`$term2`", etc. containing a varlist for each term in the model. `destest` runs through these terms, finds the terms corresponding with the termlist, and runs `termpar` with the varlist. If these global variables have not been defined, `destest` will do nothing. These global variables can of course also be used separately in `testparm`, `sw`, or related programs.

Options

The options `ndec()`, `sigcut()`, `sigsym()`, `sigsep()` have the same usage as in `desrep`:

`joint`
: If the option "`joint`" is specified, `destest` will test instead whether all the effects in all the terms are jointly equal to zero.

`equal`
: If the option "`equal`" is specified, `destest` will test whether the effects of each separate term are equal. The "`joint`" and "`equal`" options may be combined to test whether all effects are jointly equal, although this would be a somewhat peculiar hypothesis.

`ndec(#)`
: Specifies the number of decimal places. Default=3.

`sigcut(`*numlist*`)`
: `destest` places a symbol next to the Chi-square or F statistics to indicate whether these

are significant at a certain level. The `sigcut` option is used to specify these levels of significance. The *numlist* should contain a list of values in descending order with the same number of elements as the string list in `sigsym`. For example, `sigcut(.1 .05 .01 .001)` together with `sigsym(# * ** ***)` will use the symbols "#" for p < .1, "*" for p < .05, "**" for p < .01, and "***" for p < .001. The default is (.05 .01)

`sigsym(`*list*`)`
A set of symbols corresponding with the levels of significance given by `sigcut`. Default=`(* **)`.

`sigsep(#)`
The number of spaces between statistics and symbols indicating significance. Default=0.

The following two options apply only if "`using`" has been specified to write the data to a tab-delimited ascii file:

`outraw`
If `outraw` is specified then the results are written with their default formats, e.g. `%9.0g` for floats. In addition, a tab will be inserted between coefficients and significance symbols. Otherwise, the variables are written with a fixed number of decimal places as specified by the `ndec` option (default 3) and significance symbols are appended to coefficients if `sigsep`=0. Default= `nooutraw`.

`replace`
Overwrite any existing output file. If not specified, `destest` appends a number to the filename if it already exists. If no valid name has been found after appending 1 to 20, the process stops and the output is not saved. Default= `noreplace`.

Global macro variables can be used to specify different defaults for these options, either for the session or for all Stata sessions, by placing the global variables in the users profile.do.

```
$D_NDEC
$D_SIGCUT
$D_SIGSYM
$D_SIGSEP
$D_RAW
$D_REPL
```

Options specified in the `destest` command string will override these global variables.

Note

The Stata version of `desmat` was derived from a SAS macro by the same name that I wrote during the course of my Ph.D. dissertation (Hendrickx 1994). The SAS version is available at http://baserv.uci.kun.nl/~johnh/desmat/sas/.

References

Bock, R. D. 1975. *Multivariate statistical methods in behavioral research.* New York: McGraw-Hill.

Finn, J. D. 1974. *A general model for multivariate analysis.* New York: Holt, Rinehart and Winston, Inc.

Hendrickx, J. (1992). Using SAS macros and PROC IML to create special designs for Generalized Linear Models. Pp. 634- 655 in SAS Institute, *SEUGI '92. Proceedings of the SAS European Users Group International Conference*, May 19-22, 1992.

——. 1994. *The analysis of religious assortative marriage. An application of design matrix techniques for categorical models.* Dissertation Nijmegen University. Amsterdam: Thesis Publishers.

——. 1999. dm73: Using categorical variables in Stata. *Stata Technical Bulletin* 52: 2-8. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, pp. 51-59.

——. 2000. dm73.1: Contrasts for categorical variables: update. *Stata Technical Bulletin* 54: 7. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, pp. 60-61.

——. 2001. dm73.2: Contrasts for categorical variables: update. *Stata Technical Bulletin* 59: 2-5.

Knoke, D. and P. J. Burke. 1980. *Loglinear models.* Beverly Hills: Sage Publications.

Long, J. S. 1984. Estimable functions in log-linear models. *Sociological Methods & Research* 12: 339-342.

Contact information

John Hendrickx
Management Studies Group
Wageningen University
Hollandseweg 1
6706 KN Wageningen
The Netherlands

e-mail: John_Hendrickx@yahoo.com

The latest version of `desmat` is available at SSC-IDEAS:
http://ideas.uqam.ca/ideas/data/bocbocode.html.