Technical materials

Technical materials I

sheafcoef and propcnsreg: Stata modules for fitting a measurement model with causal indicators

Both chapters 3 and 5 used the `propcnsreg` package, but for subtly different purposes. In chapter 3 information from several educational category dummy variables were combined into a single optimally-scaled education variable, while in chapter 5 tested whether the relative sizes of the effect of several parental background variables have remained constant over time. The aim of this appendix is to describe both this package and a related package: `sheafcoef` (Buis, 2009b). Both `propcnsreg` and `sheafcoef` have been implemented in Stata (StataCorp, 2007).

The models implemented in both packages can be derived from the assumption that the observed variables influence the latent variable. A common alternative assumption is that the latent variable influences the observed variables. For example, factor analysis is based on this alternative assumption. To distinguish between these two situations, some authors, following Bollen (1984) and Bollen and Lennox (1991), call the observed variables “effect indicators” when they are influenced by the latent variable, and they call the observed variables “causal indicators” when they influence the latent variable. Distinguishing between these two is important as each requires a very different strategy for recovering the latent variable and its effect. In a basic (exploratory) factor analysis, which is a model for effect indicators, one assumes that the only thing the indicators have in common is the latent variable, so any correlation between these variables must be due to the latent variable, and it is this correlation that is used to recover the latent variable. In `propcnsreg` and `sheafcoef`, which estimate models for causal indicators, the latent variable is assumed to be a weighted sum of the indicators (and optionally an error term), and the weights are estimated such that they are optimal for predicting the dependent variable. Within the models implemented in the `propcnsreg` package this turns out to be equivalent to a proportionality constraint, that is, the constraint that the relative influence of each indicator remains constant over a set of other variables, in case of Chapter 5, cohort and gender.

Models for dealing with causal indicators come in roughly three flavors: A model with “sheaf coefficients” (Heise, 1972), a model with “parametricaly weighted covariates" (Yamaguchi, 2002), and a Multiple Indicators and Multiple Causes (MIMIC) model (Hauser and Goldberger, 1971). The latter two can be estimated using `propcnsreg`, while the former can be estimated using `sheafcoef`.

I.1 Sheaf coefficient

The sheaf coefficient is the simplest model of the three. Assume we want to explain a variable y using three observed variables x_1, x_2, and x_3, and we think that x_1 and x_2 actually influence y through a latent variable η and x_3 is a control variable. Because η is a latent variable, we need to fix its origin and its unit. The origin can be fixed by setting η to 0 when both x_1 and x_2 are 0, and the unit can be fixed by setting the standard deviation of η equal to 1. The model starts with a multiple regression model, where the βs are the regression coefficients and ε is a normally distributed error term, with a mean of 0 and a standard deviation that is to be estimated:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon \tag{I.1}$$

We want to turn this into equations (I.2) and (I.3), where λ is the effect of the latent variable and the γs are the effects of the observed variables on the latent variable:

$$y = \beta_0 + \lambda\eta + \beta_3 x_3 + \varepsilon \tag{I.2}$$

$$\eta = \gamma_0 + \gamma_1 x_1 + \gamma_2 x_2 \tag{I.3}$$

We can fix the origin of η by constraining γ_0 to be 0. This way η will be 0 when both x_1 and x_2 equal 0. This leaves γ_1 and γ_2. We want to choose values for these parameters such that η optimally predicts y, and the standard deviation of η equals 1. This means that γ_1 and γ_2 are going to be a transformation of β_1 and β_2. We can start with an initial guess that γ_1 equals β_1 and γ_2 equals β_2, and call the resulting latent variable η'. This will get us closer to where we want to be, as we now have values for all parameters: γ_0=0, γ_1'=β_1, γ_2'=β_2, and λ'=1. The value for λ' is derived from the fact that that is the only value where equations (I.2) and (I.3) lead to equation (I.1). However, the standard deviation of η' will generally not be equal to 1. The standard deviation of η' can be calculated as follows:

$$\mathrm{sd}(\eta') = \sqrt{\beta_1^2 \mathrm{var}(x_1) + \beta_2^2 \mathrm{var}(x_2) + 2\beta_1\beta_2 \mathrm{cov}(x_1, x_2)}$$

We can recover η by dividing η' by its standard deviation, which means that the true values of γ_1 and γ_2 are actually $\beta_1/\mathrm{sd}(\eta')$ and $\beta_2/\mathrm{sd}(\eta')$. If we divide η' by its standard deviation, then we must multiply λ' by that same number to ensure that equations (I.2) and (I.3) continue to lead to equation (I.1). As a consequence λ will equal $\mathrm{sd}(\eta')$.

This illustrates how the following set of assumptions can be used to recover the latent variable and its effect on the dependent variable:

- the latent variable is a weighted sum of the observed variables such that the latent variable optimally predicts the dependent variable.
- a constraint that fixes the origin of the latent variable.
- a constraint that fixes the unit of the latent variable.

One possible application of the sheaf coefficient is the comparison of effect sizes of different blocks of variables. For example, we may have a block of variables representing the family situation of the respondent and another block of variables representing characteristics of the work situation and we ask ourselves whether the work situation or the family situation is more important for determining a certain outcome variable. In that case we would estimate a model with two latent variables, one for the family situation and one for the work situation, and since both latent variables are standardized their effects will be comparable.

This can be useful, but a sheaf coefficient merely reorders the information obtained from a regular regression. As a consequence, it is simply a different way of looking at the regression results, and it does not impose a testable constraint. Moreover, this model does not allow for any errors in the measurement of η, as equation (I.3) does not contain an error term.

I.2 Parametricaly weighted covariates

The model with parametricaly weighted covariates Yamaguchi (2002) builds on the model with sheaf coefficients, but allows the effect of the latent variable to change over one or more other variables. This means that equation (I.4), where the effect of η changes over x_3 will be estimated, instead of equation (I.2).

$$y = \beta_0 + (\lambda_0 + \lambda_1 x_3)\eta + \beta_3 x_3 + \varepsilon \tag{I.4}$$

If η is replaced by equation (I.3), and the origin of η is fixed by constraining γ_0 to be zero, we get:

$$\begin{aligned} y &= \beta_0 + (\lambda_0 + \lambda_1 x_3)(\gamma_1 x_1 + \gamma_2 x_2) + \beta_3 x_3 + \varepsilon \\ &= \beta_0 + (\lambda_0 + \lambda_1 x_3)\gamma_1 x_1 + (\lambda_0 + \lambda_1 x_3)\gamma_2 x_2 + \beta_3 x_3 + \varepsilon \end{aligned}$$

This means the effect of x_1 (through η) on y equals $(\lambda_0 + \lambda_1 x_3)\gamma_1$, and that the effect of x_2 (through η) on y equals $(\lambda_0 + \lambda_1 x_3)\gamma_3$. This implies the following constraint: for every value of x_3, the effect of x_1 relative to x_2 will always be $(\lambda_0 + \lambda_1 x_3)\gamma_1/(\lambda_0 + \lambda_1 x_3)\gamma_2 = \gamma_1/\gamma_2$, which is a constant. In other words, the model with parametricaly weighted covariates imposes a proportionality constraint.

This proportionality constraint can also be of substantive interest without referring to a latent variable. Consider a model where one wants to explain the respondent's education (ed) with the eduction of the father (fed) and the mother (med), and that one is interested in testing whether the relative contribution of the mother's education has increased over time. `propcnsreg` will estimate this model under the null hypothesis that the relative contributions of fed and med have remained constant over time. Notice that the effects of fed and med are allowed to change over time, but the effects of fed and med are constrained to change by the same proportion over time. So if the effect of fed drops by 10% over a decade, then so does the effect of med.

The default way in which `propcnsreg` will identify the unit of the latent variable is by setting its standard deviation to 1. Alternatively, the unit can be identified in one of the following two ways: the coefficient λ_0 can be set to 1, which means that γ_1 and γ_2 represent the indirect effects of x_1 and x_2 through the latent variable on y when x_3 equals 0. This is the default parametrization, but can also be explicitly requested by specifying the `lcons` option. Alternatively, either the coefficient γ_1 or γ_2 can be set to 1, which means that the unit of the latent variable will equal the unit of x_1 or x_2 respectively. This can be done by specifying the `unit(`*varname*`)` option.

I.3 MIMIC

The MIMIC model builds on the model with parametricaly weighted covariates by assuming that the latent variable is measured with error. This means that the following model is estimated:

$$y = \beta_0 + (\lambda_0 + \lambda_1 x_3)\eta + \beta_3 x_3 + \varepsilon_y \quad \text{(I.5)}$$

$$\eta = \gamma_1 x_1 + \gamma_2 x_2 + \varepsilon_\eta \quad \text{(I.6)}$$

Where ε_y and ε_η are independent normally distributed error terms with means zero and standard deviations that need to be estimated. By replacing η in equation (I.5) by equation (I.6) one can see that the error term of this model is:

$$\varepsilon_y + (\lambda_0 + \lambda_1 x_3)\varepsilon_\eta$$

This combined error term will also be normally distributed, as the sum of two independent normally distributed variables is itself also normally distributed. The mean of this combined error term will be zero and it will have the following standard deviation:

$$\sqrt{\text{var}(\varepsilon_y) + (\lambda_0 + \lambda_1 x_3)^2 \text{var}(\varepsilon_\eta)}$$

So the empirical information that is used to separate the standard deviation of ε_y from the standard deviation of ε_η, is the changes in the residual variance over x_3. The data will thus contain rather indirect information that can be used for estimating this model. However, if the model is correct, it will make it possible to control for measurement error in the latent variable.

There is an important downside to this model, and that is that heteroscedasticity, and in particular changes in the variance of ε_y over x_3, could have a distorting influence on the parameter estimates of λ_0 and λ_1. Consider again the example of wanting to explain the respondent's education through the education of the father and the mother, but now assume that we are interested in how the effect of the latent parental education variable changes over time. In this case we have good reason to suspect that the variance of ε_y will also change over time: education consists of a discrete number of categories, and in early cohorts most of the respondents tend to cluster in the lowest categories. Over time the average level of education tends to increase, which in practice means that the respondents tend to cluster less in the lowest category, and have more room to differ from one another. As a consequence the residual variance is likely to have increased over cohorts. Normally this heteroscedasticity would not be an issue of great concern, but in a MIMIC model this heteroscedasticity is incorrectly interpreted as indicating that there is measurement error in the latent variable representing parental education. Moreover, this 'information' on the measurement error is used to 'refine' the estimates of λ_0 and λ_1. So, this would be an example where the MIMIC model would not be appropriate.

I.4 Maximization of the likelihood function

A difficulty with both the model with parametricaly weighted covariates and the MIMIC model is that the parameters are highly correlated, thus making it difficult for the standard maximization algorithms to find the maximum of the likelihood function. To overcome this issue, an EM algorithm is first used to find suitable starting values. The EM algorithm breaks the correlation by first treating the weights for the observed variables as fixed and estimating the effect of the latent variable, and then treating the effect of the latent variable as fixed and estimating the weights. By default, this is iterated 20 times or until convergence. These parameter estimates are then used as starting values for the regular maximum likelihood algorithm.

I.5 Example

The `sheafcoef` programme uses the fact that a sheaf coefficient is simply a transformation of regression coefficients, which allows it to be implemented as a post-estimation programme. This means that one must first estimate a regression model, using an estimation command like `regress` or `logit`, and then one can use `sheafcoef` to redisplay the results as a model with sheaf coefficients. It is therefore possible to use `sheafcoef` for continuous, ordered, and binomial dependent variables.

The use of this command can be illustrated using the `nlsw88.dta` dataset that comes with Stata (StataCorp, 2007). The first step is to open that dataset using the `sysuse` command, and prepare the variables.

```
. sysuse nlsw88, clear
(NLSW, 1988 extract)
.
. gen highschool = grade == 12 if grade < .
(2 missing values generated)
. gen somecollege = grade > 12 & grade < 16 if grade < .
(2 missing values generated)
. gen college = grade >= 16 if grade < .
(2 missing values generated)
.
. gen lnwage = ln(wage)
.
. gen ttl_exp2 = ttl_exp^2
.
. gen white = race == 1 if race < .
. gen other = race == 3 if race < .
```

In this example we have a set of dummies representing an individuals education (*highschool*, *somecollege*, and *college*, meaning that the reference category is those that have not finished high school), and a set of dummies representing an individual's race (*white*, and *other*, with African Americans as reference category), and we wonder which set of variables is more important for predicting an individuals wage while controlling for total experience in the labor market (*ttl_exp* and *ttl_exp2*). So, first a regression of all these variables on log wage is estimated. After that, `sheafcoef` is used, specifying in the `latent()` option the blocks of variables that belong to the same latent variable. The blocks are separated using a semi-colon (`;`). Each block of variables is preceded by its name followed by a colon (`:`). So in this example, the block of education dummies is given the name `educ` and the block of race dummies is given the name `race`. The parameters `educ` and `race` in the `main` equation represent the effects of the two latent variables. The parameters in the `on_educ` and `on_race` equations represent the effects of the dummies on the education and race latent variable respectively. The results show that education is more important than race for determining a person's income.

```
. reg lnwage white other ttl_exp ttl_exp2 highschool somecollege college

      Source |       SS       df       MS              Number of obs =    2244
-------------+------------------------------           F(  7,  2236) =  120.98
       Model |  203.545105     7  29.0778722           Prob > F      =  0.0000
    Residual |  537.417694  2236  .240347806           R-squared     =  0.2747
-------------+------------------------------           Adj R-squared =  0.2724
       Total |  740.962799  2243  .330344538           Root MSE      =  .49025

------------------------------------------------------------------------------
      lnwage |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
       white |    .118158   .0241501     4.89   0.000     .0707991    .1655169
       other |   .1079395   .0987438     1.09   0.274    -.0856996    .3015786
     ttl_exp |   .0616495    .009803     6.29   0.000     .0424255    .0808734
    ttl_exp2 |  -.0008656    .000395    -2.19   0.029    -.0016403   -.0000909
  highschool |   .1087398   .0320975     3.39   0.001     .0457958    .1716838
 somecollege |   .3568001   .0365223     9.77   0.000     .2851789    .4284213
     college |   .5167365   .0360893    14.32   0.000     .4459644    .5875086
       _cons |   .9272152    .061262    15.14   0.000      .807079    1.047351
------------------------------------------------------------------------------

. sheafcoef, latent(educ: highschool somecollege college ; race: white other)
------------------------------------------------------------------------------
      lnwage |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
main         |
        educ |   .1898757   .0107139    17.72   0.000     .1688769    .2108746
        race |   .0517396   .0105663     4.90   0.000       .03103    .0724491
     ttl_exp |   .0616495    .009803     6.29   0.000     .0424359     .080863
    ttl_exp2 |  -.0008656    .000395    -2.19   0.028    -.0016399   -.0000913
       _cons |   .9272152    .061262    15.14   0.000      .807144    1.047286
-------------+----------------------------------------------------------------
on_educ      |
  highschool |   .5726894   .1645151     3.48   0.000     .2502457    .8951332
 somecollege |   1.879124   .1570053    11.97   0.000       1.5714    2.186849
     college |   2.721446   .1063338    25.59   0.000     2.513036    2.929857
-------------+----------------------------------------------------------------
on_race      |
       white |   2.283707   .0197364   115.71   0.000     2.245024     2.32239
       other |   2.086208   1.859547     1.12   0.262    -1.558437    5.730853
------------------------------------------------------------------------------
```

The propcnsreg programme can estimate both models with parametricaly weighted covariates and MIMIC models. Unlike the models with sheaf coefficients, these models need to be separately estimated, and can thus not be as flexibly implemented as the post-estimation command sheafcoef. In particular, propcnsreg can only be used for continuous dependent variables with (approximately) normally distributed errors.

The use of propcnsreg can be illustrated by continuing the example. Now we assume that the effect of education changes for different levels of experience. The parameters in the 'constrained' panel represent the scale of education, such that parameters of high school and some college represent the positions of these levels relative to less than high school (0) and college (1). These are the effects of the education dummies on the latent variable. The parameters in the panel 'lambda' represent how the effect of the latent optimally-scaled education changes when experience changes. The unconstrained panel shows the main effects of experience and the control variables. A test of the proportionality constraint is reported at the bottom of the output.

```
. propcnsreg lnwage white other ttl_exp ttl_exp2, /*
>         */ lambda(ttl_exp ttl_exp2) /*
>         */ constrained(highschool somecollege college) /*
>         */ unit(college) nolog

                                                  Number of obs   =       2244
                                                  LR chi2(10)     =     101.57
Log likelihood = -1573.1308                       Prob > chi2     =     0.0000


Constraint: [constrained]college = 1
------------------------------------------------------------------------------
      lnwage |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
unconstrai~d |
       white |   .1166583   .0240475     4.85   0.000     .0695259    .1637906
       other |   .1101377   .0981958     1.12   0.262    -.0823226    .3025981
     ttl_exp |   .0211701    .015895     1.33   0.183    -.0099836    .0523237
    ttl_exp2 |   .0006399   .0006602     0.97   0.332    -.0006541    .0019339
       _cons |   1.150775   .0859399    13.39   0.000     .9823357    1.319214
-------------+----------------------------------------------------------------
constrained  |
  highschool |   .2431708   .0550686     4.42   0.000     .1352384    .3511032
 somecollege |   .7056825   .0538163    13.11   0.000     .6002046    .8111605
     college |          1          .        .       .            .           .
-------------+----------------------------------------------------------------
lambda       |
     ttl_exp |   .1079688   .0299633     3.60   0.000     .0492419    .1666957
    ttl_exp2 |  -.0039162   .0012162    -3.22   0.001    -.0062999   -.0015325
       _cons |  -.1390864   .1748364    -0.80   0.426    -.4817595    .2035867
-------------+----------------------------------------------------------------
ln_sigma     |
       _cons |  -.7178998    .014927   -48.09   0.000    -.7471563   -.6886434
------------------------------------------------------------------------------
LR test vs. unconstrained model: chi2(4) =     13.31   Prob > chi2 =     0.010
```

A MIMIC model can be estimated using `propcnsreg` by specifying the `mimic` option. This means that an extra parameter (`ln_sigma_latent`) is estimated representing the log of the standard deviation of the measurement error of the latent variable. In this case this does not lead to major changes in the results.

```
. propcnsreg lnwage white other ttl_exp ttl_exp2, /*
>         */ lambda(ttl_exp ttl_exp2) /*
>         */ constrained(highschool somecollege college) /*
>         */ unit(college) mimic nolog

                                                  Number of obs   =       2244
                                                  LR chi2(10)     =     135.26
Log likelihood = -1571.1459                       Prob > chi2     =     0.0000


Constraint: [constrained]college = 1
------------------------------------------------------------------------------
      lnwage |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
unconstrai~d |
       white |   .1187835   .0240303     4.94   0.000     .0716851    .1658819
       other |   .1017959   .0970624     1.05   0.294    -.0884429    .2920346
     ttl_exp |    .016967   .0151072     1.12   0.261    -.0126427    .0465766
    ttl_exp2 |   .0009142   .0006198     1.48   0.140    -.0003006    .0021291
       _cons |   1.160823   .0805208    14.42   0.000     1.003005    1.318641
-------------+----------------------------------------------------------------
constrained  |
  highschool |   .2304431   .0556791     4.14   0.000     .1213141    .3395721
 somecollege |   .7022772   .0550723    12.75   0.000     .5943375    .8102169
     college |          1          .        .       .            .           .
-------------+----------------------------------------------------------------
lambda       |
     ttl_exp |   .1182191   .0291939     4.05   0.000     .0610002    .1754381
    ttl_exp2 |    -.00456   .0011831    -3.85   0.000    -.0068787   -.0022412
       _cons |   -.160952    .168419    -0.96   0.339    -.4910472    .1691433
-------------+----------------------------------------------------------------
ln_sigma     |
       _cons |  -.8121788   .0501504   -16.19   0.000    -.9104719   -.7138858
-------------+----------------------------------------------------------------
ln_sigma_l~t |
       _cons |  -.9502158   .2506895    -3.79   0.000    -1.441558   -.4588735
------------------------------------------------------------------------------
```

This example illustrates how to estimate models with the three types of causal indicators using the `sheafcoef` and `propcnsreg` modules in Stata. A complete description of the syntax of `sheafcoef` and `seqlogit` is given below.

I.6 Syntax and options

Syntax of `sheafcoef`

```
sheafcoef, latent(varlist_1 [ ; varlist_2 [; varlist_3 [...]]]  )
    [ eform post iterate(#) level(#) ]
```

Options of `sheafcoef`

`latent(`*varlist_1* `[;`*varlist_2* `[;` *varlist_3* `[...]]]])` specifies the blocks of variables that make up the latent variables, whereby each block is separated by a semicolon (;). Each block needs to consist of at least two variables. These variables must be explanatory variables in the estimation command preceding `sheafcoef`, and the same variable can only appear in one block.

`eform` specifies that the effects of the latent variable and the control variables are exponentiated. The effects of the indicator variables in each block on its latent variable are not exponentiated, because these represent the effects of these variables on the standardized latent variable and not on the dependent variable. This option can be useful after commands like `logit` or `poisson`, as this will cause the effects on the dependent variables to be displayed in the form of odds ratios and incidence rate ratios respectively.

`post` causes `sheafcoef` to behave like a Stata estimation (e-class) command. When `post` is specified, `sheafcoef` will post the vector of transformed estimators and its estimated variance-covariance matrix to `e()`. This option, in essence, makes the transformation permanent. Thus you could, after posting, treat the transformed estimation results in the same way as you would treat results from other Stata estimation commands. For example, after posting, you could use `test` to perform simultaneous tests of hypotheses on linear combinations of the transformed estimators.

Specifying `post` clears the previous estimation results, which can then only be recovered by refitting the original model or by storing the estimation results before running `sheafcoef` and then restoring them; see [R] **estimates store**[1].

`level(#)` specifies the confidence level, as a percentage, for confidence intervals. The default is `level(95)` or as set by `set level`, see [R] **level**.

`iterate(#)` specifies the maximum number of iterations used to find the optimal step size in calculating numerical derivatives of the transformations with respect

[1]I am following Stata's convention when referencing to the manuals of Stata. These conventions are discussed in the User's Guide that comes with Stata, section 1.2.2: [U] **1.2.2 Cross-referencing**.

to the original parameters. By default, the maximum number of iterations is 100, but convergence is usually achieved after only a few iterations. You should rarely have to use this option.

Syntax of `propcnsreg`

`propcnsreg` *depvar* [*indepvars*] [*if*] [*in*] [*weight*] `,`

`constrained(`*varlist*`)` `lambda(`*varlist*`)` [`standardized` `lcons`

`unit(`*varname*`)` `mimic` `robust` `cluster(`*varname*`)` `level(#)`

em_maximize_options *maximize_options*]

Options of `propcnsreg`

`constrained(`*varlist_c*`)` specifies the variables that are measurements of the same latent variable. The effects of these variables are to be constrained to change by the same proportion as the variables specified in `lambda()` change.

`lambda(`*varlist_l*`)` specifies the variables along which the effects of the latent variable changes.

`standardized` specifies that the unit of the latent variable is identified by constraining the standard deviation of the latent variable to be equal to 1. This is the default parametrization.

`lcons` specifies that the parameters of the variables specified in the option `constrained()` measure the indirect effect of these variables through the latent variable on the dependent variable when all variables specified in the option `lambda()` are zero.

`unit(`*varname*`)` specifies that the scale of the latent variable is identified by constraining the unit of the latent variable to be equal to the unit of *varname*. The variable *varname* must be specified in `constrained()` option.

`mimic` specifies that a MIMIC model is to be estimated.

`robust` specifies that the Huber/White/sandwich estimator of variance is to be used instead of the traditional calculation; see [U] **23.14 Obtaining robust variance estimates**. `robust` combined with `cluster()` allows observations which are not independent within cluster (although they must be independent between clusters).

`cluster(`*clustervar*`)` specifies that the observations are independent across groups (clusters) but not necessarily within groups. *clustervar* specifies to which group each observation belongs; e.g., `cluster(`*personid*`)` in data with repeated observations on individuals. See [U] **23.14 Obtaining robust variance estimates**. Specifying `cluster()` implies `robust`.

`level(#)` specifies the confidence level, in percent, for the confidence intervals of the coefficients; see [R] **level**.

em_maximize_options

`emiterate(#)` specifies the maximum number of iterations for the EM algorithm. When the number of iterations equals `emiterate()`, the EM algorithm stops. If convergence is declared before this threshold is reached, it will stop when convergence is declared. The default value of `emiterate()` is 20.

`emtolerance(#)` specifies the tolerance for the coefficient vector. When the relative change in the coefficient vector from one iteration to the next is less than or equal to `emtolerance()`, the `emtolerance()` convergence criterion is satisfied. `emtolerance(1e-6)` is the default.

`emltolerance(#)` specifies the tolerance for the log likelihood. When the relative change in the log likelihood from one iteration to the next is less than or equal to `emltolerance()`, the `emltolerance()` convergence is satisfied. `emltolerance(1e-7)` is the default

These options are seldom used.

maximize_options

`difficult`, `technique(`*algorithm_spec*`)`, `iterate(#)`, `trace`, `gradient`, `showstep`, `hessian`, `shownrtolerance`, `tolerance(#)`, `ltolerance(#)`, `gtolerance(#)`, `nrtolerance(#)`, `nonrtolerance(#)`; see [R] **maximize**. These options are seldom used.