Technical materials II

seqlogit: Stata module for fitting a sequential logit model

II.1 Introduction

Chapters 6 and 7 propose two extensions to the sequential logit model, both of which have been implemented in Stata (StataCorp, 2007) as the `seqlogit` package (Buis, 2007b). The aim of this appendix is to show how to use this package. This will be done by presenting an example analysis using data already present in Stata and by giving a complete description of its syntax.

II.2 Example

The use of the `seqlogit` package is illustrated using the `nlsw88.dta` dataset that comes with Stata, and can be opened using the `sysuse` command. This dataset contains a variable *grade* measuring the respondent's highest achieved level of education in years. The dependent variable is created by transforming the variable *grade* into the variable *ed*, which measures the respondent's highest achieved level of education in the categories: less than high school (1), high school (2), some college (3), and college (4). In the example I assume that the respondents achieved their level of education by passing or failing the following sequence of transitions:

1. respondents either finished high school or not
2. those respondents that finished high school either went to college or not
3. those respondents that went to college either finished a four-year course or not

This decision tree is fed into `seqlogit` using the `tree()` option. Within this option the levels are represented with the values in the dependent variable (*ed*), the transitions are separated using commas, and the choices within a transition are separated by a colon. This tree would thus be represented in the following way: `tree(1 : 2 3 4 , 2 : 3 4 , 3 : 4)`. So the first transition is a choice between less than high school (1) and all other levels (2, 3, and 4), the second transition is a choice between leaving after high school (2) versus going to college (3 and 4), and the final transition is a choice between some college (3) and a four-year course (4)[1].

The key explanatory variable in this example is whether or not the respondent is white (*white*), and the effect of this variable can change over time (*byr*). These variables need to be specified in the `ofinterest()` and `over()` options respectively in order to make use of the post-estimation commands that come with `seqlogit`. This will cause `seqlogit` to make a new variable *_white_X_byr*, the interaction term between *white* and *byr*, and to add the variables *white* and *_white_X_byr* to the list of explanatory variables. Notice that the main effect of *byr* is not added automatically and needs to be added separately as one of the independent variables. This makes it possible for the main effect of *byr* to have a different functional form than the interaction effect. The values assigned to each level of education are specified in the `levels` option. This won't influence the output obtained from `seqlogit`, but will influence post-estimation commands like `predict` and `seqlogitdecomp`. Finally, I added a variable indicating whether or not the respondent lived in the south of the USA (*south*) as a control variable, and I added the `or` option to specify that the odds ratios are to be displayed. Together this resulted in the following sequence of commands and output:

```
. sysuse nlsw88, clear
(NLSW, 1988 extract)

. gen ed = cond(grade< 12, 1, ///
>          cond(grade==12, 2, ///
>          cond(grade<16,3,4))) if grade < .
(2 missing values generated)

. gen byr = (1988-age-1950)/10

. gen white = race == 1 if race < .
```

(*Continued on next page*)

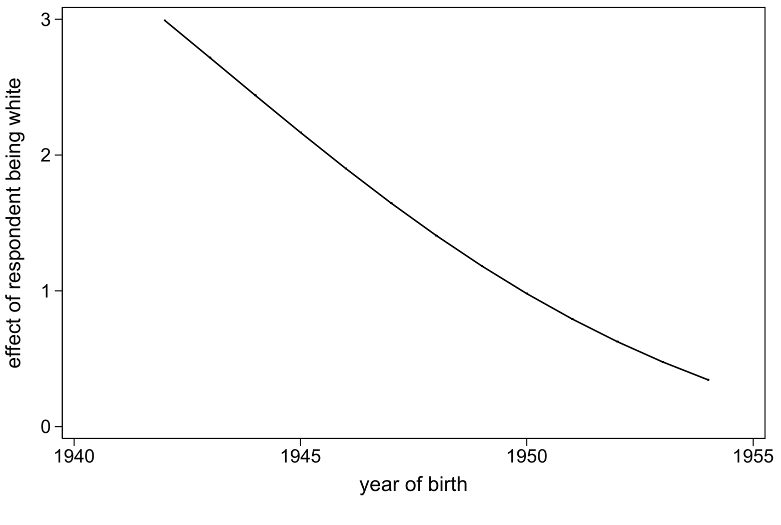
[1]The choices specified in the `tree()` option do not have to be binary (pass or fail). For example, we may believe that after finishing high school, students choose between leaving the schooling system, junior college, and college. In that case the `tree()` option would look like `tree(1 : 2 3 4, 2 : 3 : 4)`.

```
. seqlogit ed byr south,                    ///
>          ofinterest(white) over(byr)      ///
>          tree(1 : 2 3 4, 2 : 3 4, 3 : 4) ///
>          levels(1=6, 2=12, 3=14, 4= 16)  ///
>          or nolog

Transition tree:

Transition 1: 1 : 2 3 4
Transition 2: 2 : 3 4
Transition 3: 3 : 4

Computing starting values for:

Transition 1
Transition 2
Transition 3

                                                  Number of obs   =       2244
                                                  LR chi2(12)     =     110.38
Log likelihood = -2881.2013                       Prob > chi2     =     0.0000

------------------------------------------------------------------------------
          ed | Odds Ratio   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
_2_3_4v1     |
         byr |   3.377124   1.062584     3.87   0.000     1.822735    6.257061
       south |   .6440004   .0807557    -3.51   0.000     .5036723    .8234254
       white |    2.17841   .3029219     5.60   0.000     1.658726    2.860913
_white_X_byr |   .3330505   .1351488    -2.71   0.007     .1503489    .7377681
-------------+----------------------------------------------------------------
_3_4v2       |
         byr |   1.139388   .3722391     0.40   0.690     .6005969    2.161523
       south |   .8258418   .0793651    -1.99   0.046     .6840607     .997009
       white |   1.090765   .1244936     0.76   0.447     .8721274    1.364214
_white_X_byr |   .9148277   .3372712    -0.24   0.809     .4441455    1.884314
-------------+----------------------------------------------------------------
_4v3         |
         byr |   1.217693   .5757529     0.42   0.677     .4820255    3.076134
       south |   1.501026   .2063442     2.95   0.003     1.146501    1.965178
       white |   1.340784   .2183215     1.80   0.072     .9744438     1.84485
_white_X_byr |   .7585029   .4037806    -0.52   0.604     .2671958    2.153203
------------------------------------------------------------------------------
```

The results show that being white was particularly beneficial at the first transition (whether or not to finish high school), but had little effect at the higher transition. The effect of being white decreased only at the first transition. Chapter 6 showed that one can also derive the effect on the highest achieved level of education from this sequential logit model if we can assign a value to each level of education. Within Stata, this effect can be recovered after `seqlogit` using the `predict` command with the `effect` option. In this example, the levels are given values that approximately equal the years of education. One characteristic of this effect is that it will change when any of the explanatory variables change, so in order to show how the effect of `white` changed over time we first need to create a dataset where people only differ with respect to time. This is done in the example below by setting *south* and *white* to 0. The `preserve` and `restore` commands are used to ensure that these changes are only temporary. This sequence of code results in Figure II.1, which shows that the advantage of being white dropped from almost 3 years to about .5 years.

```
. preserve

. replace white = 0
(1637 real changes made)

. replace _white_X_byr = white * byr
(1479 real changes made)

. replace south = 0
(942 real changes made)

.
. predict eff, effect

.
. gen coh = byr *10 + 1950

. label variable coh "year of birth"

.
. twoway line eff coh, sort                          ///
>        ylab(0(1)3)                                 ///
>        ytitle("effect of respondent being white")

. restore
```

Figure II.1: Effect of the respondents being white on their highest achieved level of education



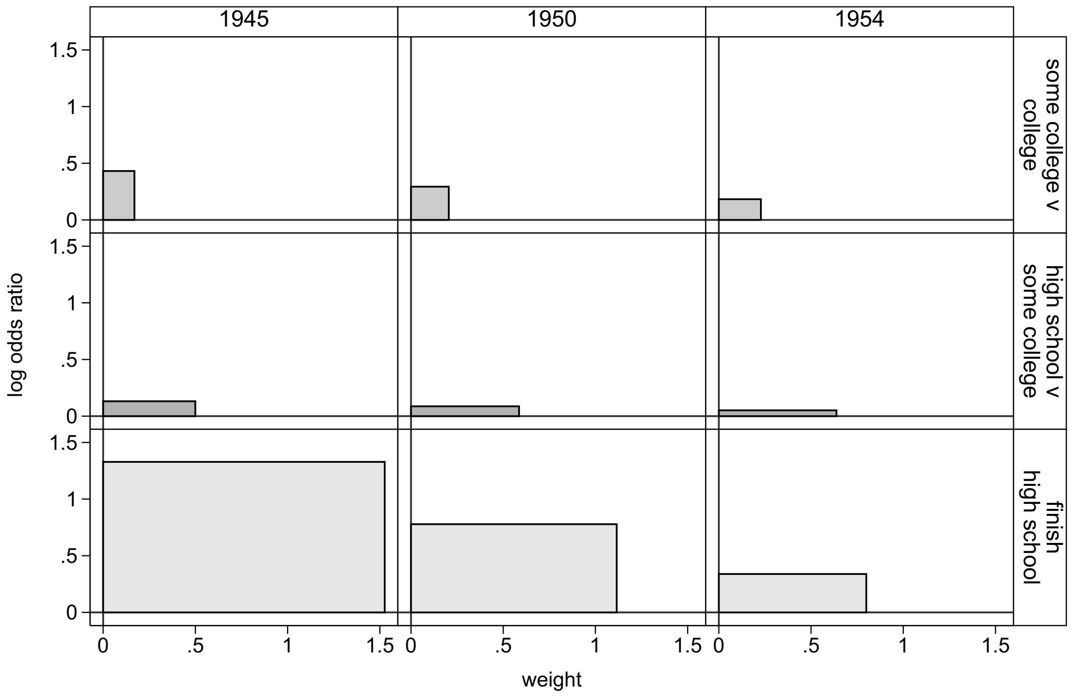
Chapter 6 showed that this effect on the highest achieved level of education is a weighted sum of the effects on passing each transition. The contribution of each transition can thus be visualized by the area of a rectangle with a width equal to the weight and a height equal to the effect on the probability of passing the transition (the log odds ratio). This is shown in Figure II.2 for three different cohorts, showing that the contribution of the first transition to the effect on the highest attained level of education has dropped dramatically over time. This graph was made using the call to `seqlogitdecomp` command shown below. The `at()` option tells that the effect is being decomposed for black respondents who are not from the south. The `overat()` option tells that this decomposition is shown for the cohorts -.5, 0, and .4. Time is measured in decades since 1950, and the `subtitle()` option is used to give more meaningful column titles. The transitions are labelled using the `eqlabel()` option. Each transition label spans two lines. This is achieved by surrounding each line with double quotes (`"  "`). Each transition's label is in turn surrounded by so-called compound quotes (`` `"  "' ``), to tell Stata which lines belong together.

```
. seqlogitdecomp, at(south 0 white 0)              ///
>          overat(byr -.5, byr 0, byr .4)          ///
>          subtitle("1945" "1950" "1954")          ///
>          eqlabel(`""finish" "high school""'      ///
>                  `""high school v" "some college""' ///
>                  `""some college v" "college""')  ///
>          xline(0) yline(0)
```

Chapter 6 also showed that the weights can in turn be decomposed as the product of three elements: The proportion of respondents at risk, the variance of the dummy variable indicating whether one passes a transition or not, and the expected gain in level of education resulting from passing. The weights and each of these elements can also be recovered using `predict` and can be displayed using the same tricks as were used when displaying the effect of `white` on the highest achieved level of education.

Chapter 7 proposed a way of assessing how sensitive the results of a sequential logit model is to unobserved heterogeneity. This strategy consists of estimating the effects of the observed variables given various scenarios concerning the degree of unobserved heterogeneity. The unobserved heterogeneity is assumed to be the result of a normally distributed unobserved variable, and the degree of unobserved heterogeneity is captured by the standard deviation of this variable. This is implemented in the `seqlogit` package in the form of the `sd()` option, which sets the standard deviation of the unobserved variable. The entire sensitivity analysis consists of multiple models, each with a different degree of unobserved heterogeneity. In the example below, just one such model is shown. Notice that we first need to drop the *_white_X_byr* variable, as this variable will be created by each call to `seqlogit`, and this will cause an error if the variable already exists.

Figure II.2: Decomposition of effect of the respondents being white on their highest achieved level of education



```
. drop _white_X_byr

. seqlogit ed byr south,                    ///
>          ofinterest(white) over(byr)      ///
>          tree(1 : 2 3 4, 2 : 3 4, 3 : 4) ///
>          or sd(1) nolog

Transition tree:

Transition 1: 1 : 2 3 4
Transition 2: 2 : 3 4
Transition 3: 3 : 4

Computing starting values for:

Transition 1
Transition 2
Transition 3
```

(*Continued on next page*)

```
                                                  Number of obs   =       2244
                                                  LR chi2(12)     =     109.61
Log likelihood = -2881.5827                       Prob > chi2     =     0.0000

------------------------------------------------------------------------------
          ed | Odds Ratio   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
_2_3_4v1     |
         byr |   4.143983   1.527877     3.86   0.000     2.011787    8.535991
       south |   .6132818   .0872595    -3.44   0.001     .4640328    .8105343
       white |   2.429149   .3848478     5.60   0.000     1.780734    3.313669
_white_X_byr |   .2739024   .1274384    -2.78   0.005     .1100418    .6817639
-------------+----------------------------------------------------------------
_3_4v2       |
         byr |   1.387615   .5412719     0.84   0.401     .6460077    2.980576
       south |   .7622253   .0879274    -2.35   0.019     .6079838    .9555969
       white |   1.212081   .1658456     1.41   0.160     .9269668     1.58489
_white_X_byr |   .7634598   .3365498    -0.61   0.540     .3217791      1.8114
-------------+----------------------------------------------------------------
_4v3         |
         byr |   1.481788     .82091     0.71   0.478     .5002889    4.388857
       south |    1.49327   .2411018     2.48   0.013     1.088189    2.049144
       white |   1.539153   .2944113     2.25   0.024     1.057944    2.239241
_white_X_byr |   .6233829   .3892371    -0.76   0.449      .183345    2.119536
------------------------------------------------------------------------------
The effect of the standardized unobserved variable is fixed at:
--------------------
equation    | sd
------------+-------
_2_3_4v1    | 1
_3_4v2      | 1
_4v3        | 1
--------------------
```

Part of the problem with unobserved variables is that the distribution of that variable tends to change over transitions due to selection. This change in distribution can be shown using the `uhdesc` command. Of particular interest is the change in the correlation between the observed variable of interest (specified in the `ofinterest()` option in `seqlogit`) and the unobserved variable. This correlation is labelled as `corr(e, x)`) in the output. It shows that over transitions an initially non-confounding variable has become a confounding variable. The syntax is similar to the `seqlogitdecomp` command.

(*Continued on next page*)

```
. uhdesc, at(south 0 white 0)                ///
>         overat(byr -.5, byr 0, byr .4)     ///
>         overlab("1945" "1950" "1954")

             | p(atrisk)    mean(e)      sd(e)  corr(e,x)
-------------+-------------------------------------------
1945         |
     2 3 4v1 |     1.000     -0.000      1.000     -0.000
       3 4v2 |     0.706      0.248      0.928     -0.070
         4v3 |     0.347      0.618      0.862     -0.075
-------------+-------------------------------------------
1950         |
     2 3 4v1 |     1.000     -0.000      1.000     -0.000
       3 4v2 |     0.815      0.161      0.945     -0.035
         4v3 |     0.414      0.530      0.876     -0.039
-------------+-------------------------------------------
1954         |
     2 3 4v1 |     1.000     -0.000      1.000     -0.000
       3 4v2 |     0.879      0.108      0.959     -0.012
         4v3 |     0.461      0.475      0.887     -0.015
```

This example illustrates the use of the `seqlogit` package and its post-estimation commands. The full syntax of these commands is described below.

II.3 Syntax and options

Syntax for `seqlogit`

`seqlogit` *depvar* [*indepvars*] [*if*] [*in*] [*weight*] `,` `tree(`*tree*`)` [
`ofinterest(`*varname*`)` `over(`*varlist*`)` `sd(#)` `rho(#)` `draws(#)`
`drawstart(#)` `or` `levels(`*levellist*`)` `constraints(`*numlist*`)`
`robust` `cluster(`*varname*`)` `nolog` `level(#)` *maximize_options*]

Options for `seqlogit`

`tree(`*tree*`)` specifies the sequence of transitions that make up the model. The transitions are separated by commas and the choices within transitions are separated by colons. The levels are represented by the levels of the *depvar*. It is thus convenient to code *depvar* as a series of integers. For example, say there are three levels, 1, 2, and 3, and the first transition consists of a choice between value 1 versus values 2 and 3, and the second transition consists (for those who didn't choose value 1) of a choice between values 2 and 3. The tree option should then be: `tree(1 : 2 3 , 2 : 3)`.

All values of *depvar* must be specified in the tree and all values in the tree must occur in *depvar*. Furthermore, all levels must be accessible through one and only one path through the tree.

`ofinterest(`*varname*`)` specifies the variable whose effect will be decomposed when using the `seqlogitdecomp` command. The variable specified is added to the list of explanatory variables.

`over(`*varlist*`)` specifies the variable(s) over which the effect of the variable specified in the `ofinterest()` option is allowed to change. This/these variable(s) and the interaction effect between the variable(s) specified in `over()` and `ofinterest()` are added to the list of explanatory variables. `ofinterest()` needs to be specified when specifying `over()`.

`sd(#)` specifies the initial standard deviation of the unobserved variable. The default is 0, which means that there is no unobserved variable.

`rho(#)` specifies the initial correlation of the unobserved variable and the variable specified in `ofinterest()`. The default is 0, which means that the unobserved variable is initially not a confounding variable.

`draws(#)` specifies the number of pseudo random draws per observation used when calculating the simulated likelihood. These pseudo random draws are created using a Halton sequence (see: [M-5] **halton()**[2]). The default is 100. Because maximum simulated likelihood is only used when the `sd()` option is specified, the `draws()` option can only be specified when the `sd()` option is specified.

`drawstart(#)` specifies the index at which the Halton sequence starts. The default is 15. This option can only be specified in combination with the `sd()` option.

`levels(`*levellist*`)` specifies the values attached to each level of the dependent variable. If it is not specified, the values of the dependent variable will be used. The syntax for *levels* is: #= #[, #= #, ...]

`or` report odds ratios

`constraints(`*numlist*`)` specifies linear constraints to be applied during estimation, see [R] **constraint**.

`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 clusters (although they must be independent between clusters).

[2]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**.

`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**.

`nolog` suppresses an iteration log of the log likelihood

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.

Syntax for `seqlogitdecomp`

`seqlogitdecomp , overat(`*overatlist*`)` [`at(`*atlist*`)`

`subtitle(`*titlelist*`)` `eqlable(`*labellist*`)` `xline(`*linearg*`)`

`yline(`*linearg*`)` `title(`*title*`))` `name(`*name* `[,` *replace*`])`

`xlabel(`*rule or values*`)` `ylabel(`*rule or values*`)`

`yscale(`*axis_suboptions*`)` `xscale(`*axis_suboptions*`)` `ysize(#)`

`xsize(#)`]

Options for `seqlogitdecomp`

Specifying the groups to be compared

`overat(`*overlist*`)` Specifies the values of the explanatory variables of the groups that are to be compared. It overrides any value specified in the `at()` option. Each comparison is separated by a comma. The syntax for *overlist* is:

varname_1 #[*varname_2* #[...]], *varname_1* #[*varname_2* #[...]], [...]

`at(`*atlist*`)` specifies the values at which the equations are evaluated. The syntax for *atlist* is: *varname_1* #[*varname_2* #...]. The equations will be evaluated at the mean values of any of the variables not specified in `at()` or `overat()`.

Say the dependent variable is highest achieved level of education, which is influenced by child's Socioeconomic Status (*ses*) and cohort (*coh*), and the interaction between *ses* and *coh* (*_ses_X_coh*). We want to compare the decomposition of the effect of *ses* over different cohorts for mean value of *ses*. Say that *coh* has only three values: 1, 2, and 3 and the mean value of *ses* is .5. Then the `overat()` and `at()` options would read:

```
overat( coh 1, coh 2, coh 3 ) at( ses .5 )
```

Notice that the values for the interaction term need not be specified in the `overat()` option, as long as it was created using the `over()` option in `seqlogit`.

Other options

`subtitle(`*titlelist*`)` specifies the titles above each group, cohort in the example above. The syntax of *titlelist* is "string" "string" [...]. The number of titles must equal the number of groups.

`eqlabel(`*labellist*`)` specifies labels for each transition. The syntax of *labellist* is "string" "string" [...]. The number of labels must equal the number of transitions.

`[x|y]line(`*numlist*`)` see: [G] **added line options**

`title(`*title*`)` see: [G] **title options**

`name(`*name* `[,` *replace*`])` see: [G] **name option**

`[y|x]scale(`*axis sub options*`)` see: [G] **axis scale options**

`[y|x]label(`*rule or values*`)` see: [G] **axis options**

`[y|x]size(#)` see: [G] **region options**

Syntax for `predict`

`predict` [`type`] *newvar* [*if*] [*in*] [`,` *statistic* `outcome(#)` `transition(#)` `choice(#)` `equation(#)`]

Options for `predict`

`transition(#)` specifies the transition, 1 is the first transition specified in the `tree` option in `seqlogit`, 2 the second, etc.

`choice(#)` specifies the choice within the transition, 0 is the first choice (the reference category), 1 the second, etc.

`equation(#)` specifies the equation, #1 is the first equation, #2 the second, etc. The "#" before the number is required.

statistic	description
`xb`	linear predictor
`stdp`	standard error of the linear predictor
`trpr`	probability of passing transition
`tratrisk`	proportion of respondents at risk of passing transition
`trvar`	variance of the indicator variable indicating whether or not the respondent passed the transition
`trgain`	difference in expected highest achieved level between those that pass the transition and those that do not
`trweight`	weight assigned to transition
`pr`	probability that an outcome is the highest achieved outcome.
`y`	expected highest achieved level
`effect`	Effect of variable of interest on expected highest achieved level. This variable is specified in the `ofinterest()` option in `seqlogit`. Interactions with the variables specified in the `over()` option of `seqlogit` are automatically taken into account.
`residuals`	difference between highest achieved level and expected highest achieved level.
`score`	first derivative of the log likelihood with respect to the linear predictor.

Syntax for `uhdesc`

`uhdesc` [`,` `at(`*atlist*`)` `overat(`*overatlist*`)` `ovarlab(`*stringlist*`)` `draws(#)`]

Options for `uhdesc`

`overat(`*overlist*`)` Specifies the values of the explanatory variables of the groups that are to be compared. It overrides any value specified in the `at()` option. Each comparison is separated by a comma. The syntax for *overlist* is:

*varname*_1 #[*varname*_2 #[...]], *varname*_1 #[*varname*_2 #[...]], [...]

`at(`*atlist*`)` specifies the values at which the equations are evaluated. The syntax for *atlist* is: *varname_1* #[*varname_2* #...]. The equations will be evaluated at the mean values of any of the variables not specified in `at()` or `overat()`.

Say the dependent variable is highest achieved level of education, which is influenced by child's Socioeconomic Status (*ses*) and cohort (*coh*) and the interaction between *ses* and *coh* (*_ses_X_coh*). We want to compare the decomposition of the effect of *ses* over different cohorts for mean value of *ses*. Say that *coh* has only three values: 1, 2, and 3 and the mean value of *ses* is .5. Then the `overat()` and `at()` options would read:

```
overat( coh 1, coh 2, coh 3 ) at( ses .5 )
```

Notice that the values for the interaction term need not be specified in the `overat()` option, as long as it was created using the `over()` option in `seqlogit`.

`overlab(`*stringlist*`)` specifies the label that is to be attached to each group specified in the `overatlist()` option. Spaces are not allowed but an "_" will be displayed as an space. The number of labels has to be the same as the number of groups specified in the `overatlist()` option.

To continue the example above: say that a value of 1 on the variable `coh` corresponds to the cohort born in 1950, a value 2 corresponds to the cohort born in 1970, a value 3 corresponds to the cohort born in 1990, then the `overlab()` option would read:

```
overlab(1950 1970 1990)
```

`draws(#)` specifies the number of pseudo random draws from the distribution of the unobserved variable used for computing the descriptive statistics. These pseudo random draws are created using a Halton sequence, see: [M-5] **halton()**. `uhdesc` uses by default the same number of draws as specified in `seqlogit`.