STATA TECHNICAL BULLETIN

January 2001

STB-59

A publication to promote communication among Stata users

Subscriptions are available from Stata Corporation, email stata@stata.com, telephone 979-696-4600 or 800-STATAPC, fax 979-696-4601. Current subscription prices are posted at www.stata.com/bookstore/stb.html.

Previous Issues are available individually from StataCorp. See www.stata.com/bookstore/stbj.html for details.

Submissions to the STB, including submissions to the supporting files (programs, datasets, and help files), are on a nonexclusive, free-use basis. In particular, the author grants to StataCorp the nonexclusive right to copyright and distribute the material in accordance with the Copyright Statement below. The author also grants to StataCorp the right to freely use the ideas, including communication of the ideas to other parties, even if the material is never published in the STB. Submissions should be addressed to the Editor. Submission guidelines can be obtained from either the editor or StataCorp.

The *Stata Technical Bulletin* (ISSN 1097-8879) is published six times per year by Stata Corporation. Stata is a registered trademark of Stata Corporation.

Contents of this issue

an74	Stata 6, Stata 7, and the STB

Patricia Branton, Stata Corporation, stata@stata.com

Stata 7 has been released, and this is the last issue of the STB that is explicitly Stata 6, which is to say, every insert in this issue will work equally well with Stata 6 or Stata 7. Future issues will contain inserts based on Stata 6 code and Stata 7 codes, and, over time, you should expect nearly all inserts to be in terms of Stata 7.

Submissions to the STB based on Stata 6 are still being accepted along with, of course, submissions based on Stata 7.

dm73.2	Contrasts for categorical variables: update

John Hendrickx, University of Nijmegen, Netherlands, J.Hendrickx@mailbox.kun.nl

Abstract: Enhancements to `desmat` and associated programs for models with categorical independent variables are described.

Keywords: contrasts, interactions, categorical variables.

The program `desmat` is used to create a design matrix for independent variables, allowing higher order interactions, and different types of contrasts (Hendrickx 1999, 2000). This update describes recent enhancements to `desmat` and its accompanying programs. The most important change to `desmat` is that it can now be used as a command prefix, similar to `xi`. When used as a command prefix, `desmat` generates a design matrix, quietly estimates the model, then calls `desrep` to present the results.

A second useful enhancement to `desmat` is that variables prefixed by an @ symbol will now be treated as continuous variables. This can be achieved now by assigning a pzat characteristic to the variable or by appending a contrast specification to the variable (Hendrickx 2000), but the new method will often be more convenient. For example, in

```
. desmat: regress weight @length rep78
```

`length` will be treated as a continuous variable, but dummies will be created for the variable `rep78`.

A third important enhancement to `desmat` is that a global macro `D_CON` can be used to specify a default contrast. This macro can be defined in the user's `profile.do` to set a new default contrast for all future Stata sessions. For example,

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

The deviation contrast with the highest category as reference will now be used as default in `desmat` models. This can be overridden by assigning a pzat characteristic to a variable, by prefixing to the variable with an @ to designate it as continuous, or by appending a contrast specification to the term.

`desmat` is still compatible with the original syntax when used as a command by itself. However, to permit new options and to make the command more in keeping with Stata syntax, the default contrast should be specified using the `defcon` option. For example,

```
. desmat vote*memb vot*educ*race educ*race*memb, dev(99)
```

Under the new syntax, this should be specified as follows, although the above syntax will still be accepted for reasons of compatibility

```
. desmat vote*memb vot*educ*race educ*race*memb, defcon(dev(99))
```

In previous versions, `desmat` printed information on variables deleted due to collinearity. This wasn't very informative, since the collinear variables were usually duplicate dummy variables produced by `desmat`. This output has now been made optional and can be requested using the `colinf` option.

Syntax for desmat

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

`desmat:` *stata_procedure* [`using` *filename*] [`if` *exp*] [`in` *range*] [*weight*] *depvar model* [`, verbose`

`defcon(`*contrast_specification*`) desrep(`*desrep_options*`)` *procedure_options*]

`fweights`, `pweights`, `aweights`, and `iweights` are allowed, and 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. All variables in the model are treated as categorical unless specified otherwise. A variable may be prefixed by an "@" to flag it as a continuous variable.

Options for desmat

When `desmat` is used as a command prefix to a Stata procedure, `if` or `in` options as well as *weight*s 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*] specifies that the results will be written to a tab-delimited ASCII file. The default extension for *filename* is `.out`. See `desrep` below for further details.

`defcon`(*contrast_specification*) can be used to specify a different contrast than the indicator contrast with the first category as reference.

Options for desmat as a command prefix

`verbose` specifies that intermediate results should be displayed. When `verbose` is not specified, `desmat` produces no output, estimates the model quietly, then calls `desrep` to display the results.

`desrep`(*desrep_options*) allows the passing of 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. An exception could be the `exp` option. `desrep` displays linear coefficients even if the procedure prints exponential coefficients, for example, the odds ratios produced by `logistic`. In such a case, one can specify

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

to display odds ratios.

Options for desmat as a 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` requests a report on which duplicate variables have been removed. 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 desrep command

`desrep` is a command 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`. `desrep` will print most model information but has not been tested for all Stata models. Note that the results of any Stata estimation procedure can be reprinted at a later point by submitting the command name with no arguments.

`desrep` has been enhanced to allow it to print z- or t-values, probabilities, and confidence intervals, as well as coefficients, standard errors, and symbols indicating significance. Global macro variables can be used to control what is printed by default. In addition, results can be written to a tab-delimited file.

Syntax for desrep

`desrep` [`using` *filename*] [`, fw(#) ndec(#) sigcut(`*numlist*`) sigsym(`*list*`) sigsep(#) nrwd(#)`

[`no`]`modinfo` [`no`]`sig` [`no`]`se zval prob ci all` [`no`]`trunc exp outraw replace`]

Options for desrep

[`using` *filename*] specifies that the results will be written to a tab-delimited ASCII file. The default extension for filename is `.out`. If *filename* already exists, `desrep` will attempt to find a valid filename by appending a number (this is done using the included `outshee2` program). The `replace` option can be used to overwrite an existing file.

`fw(#)` specifies the number of columns used to display the estimates, standard errors, and other requested statistics. The default value is 10.

`ndec(#)` specifies the number of decimal places. The default value is 3.

`sigcut(`*numlist*`)` specifies the levels of significance to be used for symbols placed next to coefficients to indicate whether these are significant at a certain level. *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)` with `sigsym(# * ** ***)` will use the symbols `#` for $p < .1$, `*` for $p < .05$, `**` for $p < .01$, and `***` for $p < .001$. The default for `sigcut` is `(.05 .01)`.

`sigsym(`*list*`)` specifies the set of symbols corresponding with the levels of significance given by `sigcut`. The default for `sigsym` is `(* **)`.

`sigsep(#)` specifies the number of spaces between coefficients and symbols indicating significance. The default is zero.

`nrwd(#)` specifies the number of columns reserved for numbering the effects. Specifying `nrwd(0)` can be used to suppress numbering. The default is 3.

[`no`]`modinfo` specifies whether or not to print information on the model and goodness of fit. The default is `modinfo`.

[`no`]`sig` specifies whether or not to display symbols for levels of significance. The default is `sig`.

[`no`]`se` specifies whether or not to print standard errors. The default is `se`.

`zval` requests printing of z values for models with a χ^2 statistic, and t values for models with an F statistic.

`prob` requests printing of p-values.

`ci` requests printing of confidence intervals.

`all` requests all standard Stata output, that is, standard errors, z or t statistics, probabilities, and confidence intervals. Specifying `all` is, thus, equivalent to specifying `zval prob ci`.

[`no`]`trunc` specifies whether or not very long labels should be cut off and the rightmost section displayed. `notrunc` will print estimates on a separate line. The default is `trunc`.

`exp` specifies that `desrep` will report multiplicative parameters, for example, incident-rate ratios in Poisson regression, odds ratios in logistic regression. Note that if `exp` is not specified, `desrep` will produce the linear estimates even if the procedure produces multiplicative versions. 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` specifies that results are written with their default formats, for example, `%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, and significance symbols are appended to coefficients if `sigsep` is zero.

`replace` specifies to 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.

Macro variables to control layout

Macro variables can be used 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_NRWD        D_CI
D_NDEC          D_SIG         D_ALL
D_SIGCUT        D_SE          D_TRUNC
D_SIGSYM        D_ZVAL        D_RAW
D_SIGSEP        D_PROB        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 "# * ** ***"
```

The showtrms command

`showtrms` produces a legend of the dummy variables produced by `desmat`, the terms these pertain to, and the contrasts used. The `showtrms` command has no options. It 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.

The destest command

`destest` is for use after estimating a model with a design matrix generated by `desmat` to perform a Wald test on model terms. Like `desrep`, `destest` can now write results to a tab-delimited file, symbols and cutpoints for levels of significance can be specified, and some aspects of the layout of results can be controlled through macro variables. In addition, `destest` now uses `svytest` if survey models have been estimated.

Syntax for destest

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

`sigsym(`*list*`) sigsep(#)`]

Options for destest

The options `ndec`, `sigcut`, `sigsym`, `sigsep`, `outraw`, and `replace` have the same usage as in `desrep`.

`joint` specifies that `destest` will test whether all the effects in all the terms are jointly equal to zero.

`equal` specifies that `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.

The global macro variables listed below can be used to specify different defaults for these options, either for only the current session or for all Stata sessions, by placing the global variables in the user's `profile.do`.

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

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

References

Hendrickx, J. 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.

dm83 | Renaming variables: changing suffixes

Stephen P. Jenkins, University of Essex, UK, stephenj@essex.ac.uk
Nicholas J. Cox, University of Durham, UK, n.j.cox@durham.ac.uk

Abstract: `rensfix` is a command to rename variables, changing the suffix. It complements `renpfix`, an official Stata command to rename variables, changing the prefix.

Keywords: `rensfix`, `renpfix`, `rename`, variable names, data management.

Syntax

`rensfix` *old_stub* [*new_stub*]

Description

`rensfix` renames all variables whose names end with *old_stub*, so that they end instead with *new_stub*. If *new_stub* is not specified, *old_stub* is removed. It, therefore, matches the existing Stata command `renpfix` (see [R] **rename**), which renames variables, changing the prefix of each name.

In contrast to `renpfix`, however, `rensfix` will not rename any variable whose name ends in the suffix specified unless all new names implied are in fact legal names for new variables. Thus, either all names implied are acceptable, or `rensfix` warns you otherwise and makes no changes.

Examples

We illustrate `rensfix` using Stata's auto data. We begin by changing the variable `rep78` to `rep`

```
. use auto
(1978 Automobile Data)
```

```
. ds
make      price     mpg       rep78     hdroom    trunk     weight    length
turn      displ     gratio    foreign
. rensfix 78
. ds
make      price     mpg       rep       hdroom    trunk     weight    length
turn      displ     gratio    foreign
```

and then change any variable name ending in `e` to end in `123`

```
. rensfix e 123
. ds
mak123    pric123   mpg       rep       hdroom    trunk     weight    length
turn      displ     gratio    foreign
```

and then show what happens if `rensfix` would result in a name of more than 8 characters.

```
. rensfix 123 12345
pric12345 invalid name
r(198);
. ds
mak123    pric123   mpg       rep       hdroom    trunk     weight    length
turn      displ     gratio    foreign
```

dm84	labjl: Adding numerical codes to value labels

Jens M. Lauritsen, County of Fyn, Denmark, jm.lauritsen@dadlnet.dk

Abstract: To document fully analyses and to assure no numerical code misinterpretations are made, it is advisable to add value labels to all categorical variables as part of the documentation of a dataset. In analysis, one often wishes to make a particular calculation, listing, or analysis based on the numerical code of the variable. To facilitate the ease of having labels as well as numerical codes visible, the routine `labjl` will add the numerical code to the label as well as remove it afterwards.

Keywords: value label, categorical value codes, documentation.

Syntax

`labjl` *varlist* [`, add remove noalign n(#)`]

Description

`labjl` adds the numeric code of a defined value label to each label for the variables in *varlist*. It can be used when a user wants to see the numeric code and the label in tables or other displays. If several variables are shown, the labels used by any of these will be changed. If a variable does not have a value label attached to it, nothing happens.

The routine can add the numerical value as well as restore the original label. The routine does not change any data values.

Options

`add` specifies that the value is to be added to the label (the default).

`remove` specifies that the value is to be removed from the label when the numerical code is equal to the first nonblank part of the label.

`noalign` specifies that the numeric codes will not be left aligned.

`n(#)` specifies that the user wants to increase the limit of codes. The default is the interval from 0 to 25.

Example

For a variable called `sex`, we use `tabulate` before and after using `labjl`.

```
. tabulate sex
        sex |      Freq.     Percent        Cum.
------------+-----------------------------------
     Female |        490       12.17       12.17
       Male |       3537       87.83      100.00
------------+-----------------------------------
      Total |       4027      100.00

. labjl sex
Numerical codes added to Label: sex
. tabulate sex
        sex |      Freq.     Percent        Cum.
------------+-----------------------------------
   1 Female |        490       12.17       12.17
   2   Male |       3537       87.83      100.00
------------+-----------------------------------
      Total |       4027      100.00
```

If one does not want to left align the codes, they can use the following:

```
. labjl sex, noalign
Numerical codes added to Label: sex
. tabulate sex
        sex |      Freq.     Percent        Cum.
------------+-----------------------------------
   1 Female |        490       12.17       12.17
     2 Male |       3537       87.83      100.00
------------+-----------------------------------
      Total |       4027      100.00
```

The numeric codes can be removed as in the next example.

```
. labjl sex, remove
Numerical codes removed from Label: sex
. tab sex
        sex |      Freq.     Percent        Cum.
------------+-----------------------------------
     Female |        490       12.17       12.17
       Male |       3537       87.83      100.00
------------+-----------------------------------
      Total |       4027      100.00
```

To add numerical codes to all variables for which value labels have been defined, one can use the code below.

```
. labjl _all /* add to all labels */
. labjl _all, remove /* remove from all labels */
```

dm85 | listjl: List one variable in a condensed form

Jens M. Lauritsen, County of Fyn, Denmark, jm.lauritsen@dadlnet.dk

Abstract: When checking datafiles during documentation it can be helpful to list id numbers in a condensed form for checking or finding of paper material. `listjl` does this for all observations or broken down by one different variable.

Keywords: data quality, id variable, observation id.

Syntax

`listjl` *variable* [`if` *exp*] [`in` *range*] [`, by(`*variable*`)`]

Description

`listjl` will list the variable indicated in a condensed format, optionally broken down by one group variable.

Options

`by(`*variable*`)` specifies that the listing is to be broken down by the specified variable. When indicated, a frequency table of this variable will be shown.

Example

Here is an example of listing observations broken down by variable `sex` for the first 50 observations:

```
. listjl id in 1/50, by(sex)
-> tabulation of sex
        sex |      Freq.     Percent        Cum.
------------+-----------------------------------
          K |          5       10.00       10.00
          M |         45       90.00      100.00
------------+-----------------------------------
      Total |         50      100.00

Group: sex =  K    id :
8 34 35 41 42

Group: sex =  M    id :
1 2 4 5 6 7 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
30 31 32 33 36 37 38 39 40 43 44 45 46 47 48 49 50 51
```

Now, an example of a listing without the `by` option but using the `if` option:

```
. listjl id if id < 52
1 2 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51
```

dm86	Sampling without replacement: absolute sample sizes and keeping all observations

Nicholas J. Cox, University of Durham, UK, n.j.cox@durham.ac.uk

Abstract: `swor` is a command for sampling without replacement which allows specification of absolute sample sizes and (optionally) keeping of nonsampled observations in memory. It complements the official Stata command `sample`.

Keywords: `sample`, sampling without replacement, data management.

Syntax

`swor` # [`if` *exp*] [`in` *range*] [, `by(`*groupvars*`)` `generate(`*newvar*`)` `keep`]

Description

`swor` draws a pseudo-random sample of size # from the data in memory, by default discarding (optionally keeping) the remaining observations. Observations not meeting the optional `if` and `in` criteria are kept (sampled at 100%). # must be positive and no greater than the number of observations in each group to be sampled.

Options

`by(`*groupvars*`)` specifies that a sample of size # is to be drawn within each set of values of *groupvars*.

`generate(`*newvar*`)` generates *newvar* containing 1 for those observations included in the sample and 0 for those not included.

`keep` overrides the default dropping of nonsampled observations. `keep` must be combined with `generate()`.

Discussion

The Stata command `sample` (see [R] **sample**) also carries out sampling without replacement, specified in percent to be sampled, rather than absolute number. See also Weesie (1997) for an extension of `sample` to sampling clusters of observations. Various bootstrap sampling commands allow sampling with replacement; see, in particular, [R] **btstrap**. Alternatively, check out the `resample` command of Gleason (1997, 1999).

Two features of `sample` are sometimes limitations. First, `sample` is designed to draw a sample as a specified percent of the observations in memory. Often the problem in practice is to draw a sample of a specified absolute size. Second, those observations not included in the sample are dropped from memory, although all observations not satisfying any `if` or `in` restrictions are kept. Sometimes it is desired to keep all nonsampled observations in memory.

`swor` is designed to meet these needs. The number to be specified is the absolute sample size, not a percent. All observations may be kept in memory using the `keep` option, in which case the sampled observations must be tagged by a new variable named in `generate()`.

If you are serious about drawing random samples, you must first set the random number seed; see [R] **generate**.

Examples

First, we set the seed for reproducibility

```
. log using sampling
. set seed 987654321
```

and then draw a sample of size 100

```
. swor 100
```

We can also keep all `sex` not equal to 0, but sample 100 from `sex` equal to 0

```
. swor 100 if sex == 0
```

or sample 100 from each group of `sex`

```
. swor 100, by(sex)
```

or sample 100 from each group of `sex`, but keep all the observations

```
. swor 100, by(sex) gen(sample) keep
```

Finally, we can sample 100 if `sex` is 0 and keep only those observations.

```
. swor 100 if sex == 0, gen(sample)
. keep if sample
```

References

Gleason, J. R. 1997. ip18: A command for randomly resampling a dataset. *Stata Technical Bulletin* 37: 17–22. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, pp. 77–83.

——. 1999. ip18.1: Update to resample. *Stata Technical Bulletin* 52: 9–10. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, p. 119.

Weesie, J. 1997. dm46: Enhancement to the sample command. *Stata Technical Bulletin* 37: 6–7. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, pp. 37–38.

sbe36.1	Summary statistics for diagnostic tests

Paul T. Seed, King's College London, UK, paul.seed@kcl.ac.uk
Aurelio Tobias, Hospital de la Santa Creu i Sant Pau, Barcelona, Spain, atobias@cocrane.es

Abstract: An extensive revision of the `diagtest` command introduced in Tobias (2000) is introduced and illustrated.

Keywords: diagnostic test, sensitivity, specificity, predictive values, confidence intervals, contingency tables.

Introduction

The new command `diagt` is a complete revision of the earlier command `diagtest` (Tobias 2000). It should be regarded as succeeding it and replacing it. Both commands are used to assess a simple diagnostic test in comparison with a reference standard (or "gold standard"), assumed to be completely accurate. The diagnostic test is generally used because it is cheaper, quicker, or less invasive than the reference standard, but may not be as reliable. Results are typically presented in a 2×2 table and summarized as four percentages: sensitivity, specificity, positive predictive values, and negative predictive values, with their respective confidence intervals. `diagti` is an immediate version of `diagt` that does not require data to be entered.

The commands `diagt` and `diagtest` differ in the following ways:

- The exact binomial distribution is used instead of the normal approximation, based on command `ci`.
- An algebraic error that caused `diagtest` to give confidence intervals that were too wide has been corrected. Essentially, `diagtest` used the grand total instead of the row or column totals when calculating the standard errors.
- Only `fweight`s are allowed, as with `ci` and `binomial`.
- The command format is changed. The outcome (the true disease status) is placed before the predictor (the diagnostic test). This makes it consistent with commands such as `logistic`, `roctab`, `cs`, and `cc`.

- The table layout is also changed. Rows represent disease status. As with commands `cc` and `cs`, rows represent true disease status, columns represent exposure (or test) result. Positive results are given in the top and left of the table, before negative ones.
- An immediate version is provided.
- The name has been changed. This is partly because the shorter name allows the immediate version to end in i, and partly to reduce the risk of people confusing the two command formats and so getting the wrong estimates.

Sensitivity is the proportion of true positives that are correctly identified by the test, and specificity the proportion of true negatives correctly identified. Typically, these are regarded as fixed properties of a particular test, independent of the prevalence of the disease. However, in medical practice, the result of the diagnostic test is all that is known. The positive predictive value (PPV) is the proportion of patients with positive test results who are correctly diagnosed. The negative predictive value (NPV) is the proportion of patients with negative test results who are correctly diagnosed. These values help a clinician trying to make a diagnosis for a particular patient.

	Test result		
True disease status	Positive(+)	Negative(−)	
Abnormal(+)	a	b	$\Longrightarrow$ Sensitivity $= a/(a+b)$
Normal(−)	c	d	$\Longrightarrow$ Specificity $= d/(c+d)$
	$\Downarrow$	$\Downarrow$	
	PPV $= a/(a+c)$	NPV $= d/(b+d)$	

Table 1. Definition of sensitivity, specificity and predictive values.

The predictive values of a test in clinical practice depend critically on the prevalence of the abnormality in the patients being tested; these values depend on the prevalence of the disease. Sometimes the prior likelihood of the current patient having the disease can be estimated from other signs and symptoms. This can be used instead of the prevalence.

The predictive values (PPV and NPV) can be calculated for any prevalence by using Bayes' theorem, as follows:

$$\text{PPV} = \frac{\text{Sensitivity} \times \text{Prevalence}}{\text{Sensitivity} \times \text{Prevalence} + (1 - \text{Sensitivity}) \times (1 - \text{Prevalence})}$$

$$\text{NPV} = \frac{\text{Specificity} \times (1 - \text{Prevalence})}{\text{Specificity} \times (1 - \text{Prevalence}) + (1 - \text{Specificity}) \times \text{Prevalence}}$$

Syntax

`diagt` *diagvar testvar* [*weight*] [`if` *exp*] [`in` *range*] [`, prev(#) level(#)` *tabulate_options*]

`diagti` $\#_a$ $\#_b$ $\#_c$ $\#_d$ [`, prev(#) level(#)` *tabulate_options*]

where *diagvar* is the variable which contains the real status of the patient, and where *testvar* is the variable which identifies the result of the diagnostic test. *testvar* and *diagvar* can have only two nonmissing values. The higher value must identify the positive result of the test or the diseased status of the patient.

For `diagti`, $\#_a$, $\#_b$, $\#_c$, and $\#_d$ are, respectively, the numbers of true positives (diseased subjects with correct positive test results), false negatives (diseased, but negative test), false positives (no disease, but positive test) and true negatives (no disease, negative test).

`fweights` are allowed.

Exact binomial confidence intervals are given, as with the command `ci`.

Options

`prev(#)` specifies the estimated prevalence, in percent, of the disease to be used in estimating the positive and negative predicted values using Bayes' theorem. If the `prev` option is used, the confidence interval is only displayed for the sensitivity and specificity values. Otherwise, the prevalence is estimated from the data.

`level(#)` specifies the confidence level, in percent, for confidence intervals. The default is `level(95)` or as set by `set level`.

All `tabulate` command options are available.

Example

The same example is considered here as in Tobias (2000) describing `diagtest`. Altman and Bland (1994a, 1994b) consider the relationship between the results of a liver scan test and the correct diagnosis (Drum and Christacapoulos 1972). The proportions that were correctly diagnosed by the scan were 89.53% for normal liver scan, and 62.79% for those with abnormal scan. The proportion of correct diagnoses among the patients with abnormal liver scan test was 87.83%, and among the patients with normal liver scans such proportion was 66.67%.

Here are summary results for the liver scan test and the correct diagnosis.

```
. diagt diag test [fw=n]
           |    diagnostic test
      true |        result
diagnostic |      Pos.       Neg. |     Total
-----------+----------------------+----------
  Abnormal |       231         27 |       258
    Normal |        32         54 |        86
-----------+----------------------+----------
     Total |       263         81 |       344
True abnormal diagnosis defined as diag = 1 (labelled Abnormal)
                                                      [95% Conf. Inter.]
------------------------------------------------------------------------
Sensitivity                     Pr( +| D)  89.53%      85.14%   92.99%
Specificity                     Pr( -|~D)  62.79%      51.70%   72.98%
Positive predictive value       Pr( D| +)  87.83%      83.26%   91.53%
Negative predictive value       Pr(~D| -)  66.67%      55.32%   76.76%
------------------------------------------------------------------------
Prevalence                      Pr(D)      75.00%      70.08%   79.49%
------------------------------------------------------------------------
```

In the liver scan study, the percentage of abnormality was 75%. If the same test was used in a different clinical setting where the prevalence of abnormality was 0.25%, we would have a positive predictive value of 44.51% and a negative predictive value of 94.74%.

Now, we show summary results for the liver scan test and the correct diagnosis for a prevalence of abnormality of 25%.

```
. diagt diag test [fw=n], prev(25)
           |    diagnostic test
      true |        result
diagnostic |      Pos.       Neg. |     Total
-----------+----------------------+----------
  Abnormal |       231         27 |       258
    Normal |        32         54 |        86
-----------+----------------------+----------
     Total |       263         81 |       344
True abnormal diagnosis defined as diag = 1 (labelled Abnormal)
                                                      [95% Conf. Inter.]
------------------------------------------------------------------------
Sensitivity                     Pr( +| D)  89.53%      85.14%   92.99%
Specificity                     Pr( -|~D)  62.79%      51.70%   72.98%
Positive predictive value       Pr( D| +)  60.15%          .%       .%
Negative predictive value       Pr(~D| -)  40.51%          .%       .%
------------------------------------------------------------------------
Prevalence                      Pr(D)      25.00%          .%       .%
------------------------------------------------------------------------
```

The same results can be achieved without any data in memory using the immediate form of the command:

```
. diagti 231 27 32 54
      True |
   disease |      Test result
    status |      Pos.       Neg. |     Total
-----------+----------------------+----------
  Abnormal |       231         27 |       258
    Normal |        32         54 |        86
-----------+----------------------+----------
     Total |       263         81 |       344
------------------------------------------------------------------------
Sensitivity                     Pr( +| D)  89.53%      85.14%   92.99%
Specificity                     Pr( -|~D)  62.79%      51.70%   72.98%
Positive predictive value       Pr( D| +)  87.83%      83.26%   91.53%
Negative predictive value       Pr(~D| -)  66.67%      55.32%   76.76%
------------------------------------------------------------------------
Prevalence                      Pr(D)      75.00%      70.08%   79.49%
------------------------------------------------------------------------
```

References

Altman, D. G. and J. M. Bland. 1994a. Diagnostic tests 1: sensitivity and specificity. *British Medical Journal* 308: 1552.

——. 1994b. Diagnostic tests 2: predictive values. *British Medical Journal* 309: 102.

Drum, D. E. and J. S. Christacapoulos. 1972. Hepatic scintigraphy in clinical decision making. *Journal of Nuclear Medicine* 13: 908–915.

Tobias, A. 2000. sbe36: Summary statistics report for diagnostic tests. *Stata Technical Bulletin* 56: 16–18.

sbe41	Ordinary case–cohort design and analysis

Vincenzo Coviello, Unità di Epidemiologia e Statistica ASL Ba/1, Italy, coviello@mythnet.it

Abstract: The case–cohort design is an efficient alternative to a full cohort analysis. Two new commands for ordinary case–cohort designs are presented. They randomly select a sample from a cohort, prepare the resulting dataset for analysis using a Cox regression model and compute the asymptotically consistent Self–Prentice variance estimator of the parameters.

Keywords: Cohort studies, nested case–control design, survival analysis, Cox regression model, variance estimation.

Introduction

Various designs have been proposed as useful alternatives to the standard full cohort analysis when data collection for any subject may be very expensive. In the case–cohort design (see Barlow et al. 1999, Clayton and Hills 1993, Langholz and Thomas 1990, 1991, and Rothman and Greenland 1998, for example), covariate information is assessed in a sample of subjects selected randomly from the entire cohort, the subcohort, and in all individuals who fail, whether they are in the subcohort or not. The case–cohort design has some advantages with respect to the nested case–control study mainly when the cohort under study is fixed (i.e., without staggered entry times), failures are rare, and there is little loss to follow-up. Unlike the nested case–control study, the case–cohort design allows one to use the same sample of subjects to analyze several failure time outcomes. Furthermore, the subcohort is chosen without regard to any time scale. Recently, simplified methods have appeared for parameter and variance estimation, thus, allowing the analysis of the case–cohort design using a Cox model suitably adapted. Here, we present two new Stata commands that assist the user in the fundamental steps of case–cohort design and analysis: `stcascoh` to sample the full cohort and prepare a dataset for analysis, and `stselpre` to calculate the variance as proposed by Self and Prentice.

Syntax for sampling cohort and preparing dataset

`stcascoh` [*varlist*] [`if` *exp*] [`in` *range*] , `alpha(`#`)` [`group(`*varnames*`)` `generate(`*varlist*`)` `eps(`#`)`
`seed(`#`)` `noshow`]

`stcascoh` is for use with survival-time data. You must `stset` the data with an `id()` variable before using this command.

Description

`stcascoh` is used to create an appropriate dataset for analysis as a case–cohort study, drawing an α-fraction random sample of the full cohort and including all failures whether they occur in the random sample or not. To this aim, `stcascoh` expands observations which fall into two parts: 1) time interval $(t_0, t-\epsilon]$, and 2) time interval $(t-\epsilon, t]$. For cases included in the subcohort, both segments are retained in the final dataset, whereas for cases not in the subcohort just the last segment is retained.

The variables in the table below are added to the dataset.

`_subco`	coded 0 for subcohort member with no failure
	coded 1 for subcohort member who failed
	coded 2 for nonsubcohort member who failed (nonsubcohort case)
`_wSelPre`	log-weights of records as in the Self–Prentice method
`_wBarlow`	log-weights of records as in the Barlow method

The names of the new variables and the sampling fraction are saved as Stata characteristics as shown in the table below.

`_dta[Subco]`	subcohort membership variable name
`_dta[wSelPre]`	Self–Prentice log-weights variable name
`_dta[wBarlow]`	Barlow log-weights variable name
`_dta[Alpha]`	sampling fraction

varlist defines variables that will be retained in the final dataset. If *varlist* isn't specified, all variables are carried over into the resulting dataset. Observations not meeting `if` and `in` criteria are dropped even if they fail. Randomness in the sampling is obtained using Stata's `uniform()` function.

Options

`alpha(#)` specifies the sampling fraction. The sampling fraction can be expressed as real or integer.

`group(`*varlist*`)` specifies that the alpha sample is to be drawn within each set of values of *varlist*, thus, maintaining the proportion in each group.

`generate(`*varlist*`)` specifies other variable names for the three generated variables.

`eps(#)` specifies a (typically small) number so that a case that is in the risk set at time t is represented in the expanded data by an "infinitesimal" episode $(t - \text{eps}, t]$. `eps` should be set to a number that is small compared to the measurement unit of time. The default value of `eps` is .001.

`seed(#)` specifies a seed for random sampling. Providing the same seed means that the same subcohort is selected each time even if the definition of the failure event or time axis has been changed in `stset`.

`noshow` prevents `stcascoh` from showing the names of the key `st` variables.

Example

As an example, consider data contained in Appendix VIII of Breslow and Day (1987) concerning workers employed in a Welsh nickel refinery. Mortality for nasal cancer and sinus cancer has been analyzed using "time since first exposure" as analysis time. A preliminary description of the full cohort data is given below.

```
. use nickel, clear
. stset dataout, f (tumnas) origin(dataass) enter(datain) id(id)
. stdes
        failure _d:  tumnas
  analysis time _t:  (dataout-origin)
            origin:  time dataass
 enter on or after:  time datain
                id:  id
                                    |-------------- per subject --------------|
Category                    total        mean         min      median         max
------------------------------------------------------------------------------
no. of subjects               679
no. of records                679           1           1           1           1
(first) entry time                    17.98332    9.359375    18.29639    36.04663
(final) exit time                     40.58158    10.04102    40.06299    75.59204
subjects with gap               0
time on gap if gap              0           .           .           .           .
time at risk            15344.223    22.59827    .3996582    21.75623    48.6543
failures                       56    .0824742           0           0           1
------------------------------------------------------------------------------
```

Now, we prepare this dataset for analysis as a case–cohort design using a sampling fraction of 0.3.

```
. stcascoh, a(0.3) seed(123)
        failure _d:  tumnas
  analysis time _t:  (dataout-origin)
            origin:  time dataass
 enter on or after:  time datain
                id:  id
Sample composition
 Subcohort |
    member |  Censored    Failure |     Total
-----------+----------------------+----------
        No |       436         39 |       475
       Yes |       187         17 |       204
-----------+----------------------+----------
     Total |       623         56 |       679
Total sample =  243
No risk set with less than 4 controls
New stset definition
               id:  id
    failure event:  _d ~= 0 & _d ~= .
obs. time interval:  (_t0, _t]
 enter on or after:  time _t0
 exit on or before:  failure
```

```
------------------------------------------------------------------------------
      260  total obs.
        0  exclusions
------------------------------------------------------------------------------
      260  obs. remaining, representing
      243  subjects
       56  failures in single failure-per-subject data
 4653.123  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =  9.373047
                                  last observed exit t =  75.59204
```

The output consists of

1. A table displaying the sample composition with respect to the full cohort. Note that from 679 subjects in the full cohort, 204 have been randomly sampled (subcohort). 39 more subjects who failed are added to the subcohort, so the total sample is 243 subjects.
2. A message declaring that in this sample no risk sets with three or fewer controls exist. Otherwise, a table illustrating risk sets with such few controls is displayed. This is a warning for the loss of efficiency of the design that may occur when the subcohort becomes small due to many failures or censorings. The sampling fraction α should be increased in such cases.
3. A new `stset` definition that fixes the entry and exit time to the present `_t`, `_t0` and `_d` variables. This is necessary because nonsubcohort cases cannot rely on the original entry times. Note that after `stcascoh`, the total analysis time is reduced to 4653.123 starting from 15344.223 in the full cohort.

Syntax for estimating the model

`stselpre` *varlist* [`if` *exp*] [`in` *range*] [`, nohr level(#) self` {`breslow` | `efron` | `exactm` | `exactp`} `noshow`]

`stselpre` is for use with survival-time data. Data must be prepared for case–cohort analysis by `stcascoh` before using this command.

Description

`stselpre` returns estimates and standard errors from proportional hazards fit to case–cohort data. Coefficients are estimated according to two methods: 1) the Self–Prentice method where risk sets use just the subcohort member at risk, and 2) the Prentice (1986) method where risk sets are augmented by nonsubcohort cases when they fail.

The asymptotic Self–Prentice method variance–covariance matrix and standard errors are computed using the simplification described in Therneau and Li (1999). The syntax of `predict` following `stselpre` is as in `stcox`.

Options

`nohr` specifies that coefficients rather than hazard ratios are to be displayed. `nohr` may be typed at estimation or when replaying previously estimated results.

`level(#)` specifies the confidence level for the confidence intervals of the coefficients. `level` may also be specified when replaying previously estimated results.

`self` specifies that the Self–Prentice method coefficient vector is to be saved in `e(b)`. By default, Prentice method coefficients are saved.

`breslow`, `efron`, `exactm`, and `exactp` each specify a method for handling tied deaths that the underlying `stcox` command executes. `efron` is the default.

`noshow` prevents `stselpre` from displaying the identities of the key `st` variables above its output.

(*Continued on next page*)

Remarks

When a dataset is prepared, the parameter estimates can be easily obtained using a Cox model. However, their variance is somewhat complicated because of the correlation between risk sets induced by the sampling. Lin and Ying (1993) and independently Barlow (1994) proposed the use of the robust variance estimator for the case–cohort design as a simpler alternative to the asymptotically consistent estimator provided by Self and Prentice (1988). Recently, Therneau and Li (1999) proved that the Self and Prentice variance estimator can be obtained by correcting the standard variance estimate with a matrix derived from a subset of dfbeta residuals. Both estimates are now available in Stata. In estimating the model, three methods can be implemented: 1) Prentice, 2) Self and Prentice, and 3) Barlow. They differ in the composition of risk sets and in the weight ascribed to the nonsubcohort case and to the subcohort member. Both aspects can be handled by preparing an appropriate dataset and by using as offset terms, the log-weights stored in the variables created by `stcascoh`.

Example (continued)

In their analysis, Breslow and Day (1987) found three significant risk factors. Their final model includes four variables: 1) $\log(\text{age at first employment} - 10)$, 2) $(\text{year first employed} - 1915)/100$, 3) $2(\text{year first employed} - 1915)/100$, and 4) $\log(\text{exposure} + 1)$. Here are the results in the full cohort:

```
. stcox afe_10 yfe_15 yfe2_15 logexp,nolog nohr
        failure _d:  tumnas
  analysis time _t:  (dataout-origin)
            origin:  time dataass
 enter on or after:  time datain
                id:  id

Cox regression -- no ties

No. of subjects =          679                     Number of obs   =       679
No. of failures =           56
Time at risk    =  15344.22327
                                                    LR chi2(4)      =     84.42
Log likelihood  =   -285.22255                      Prob > chi2     =    0.0000
------------------------------------------------------------------------------
      _t |
      _d |      Coef.   Std. Err.       z     P>|z|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  afe_10 |   2.155984   .4289709      5.026   0.000       1.315217    2.996752
  yfe_15 |  -.0882464   .3164137     -0.279   0.780      -.7084059    .5319132
 yfe2_15 |  -1.262437   .5085606     -2.482   0.013      -2.259198    -.265677
  logexp |   .7710877   .1746156      4.416   0.000       .4288474    1.113328
------------------------------------------------------------------------------
```

Now, we estimate the same model from a case–cohort design using the dataset previously prepared by `stcascoh` according to the various methods mentioned above and using the robust variance estimator. We begin with the Prentice method.

```
.stcox afe_10 yfe_15 yfe2_15 logexp, robust nohr nolog
        failure _d:  _d
  analysis time _t:  _t
 enter on or after:  time _t0
                id:  id

Cox regression -- no ties

No. of subjects =          243                     Number of obs   =       260
No. of failures =           56
Time at risk    =  4653.122862
                                                    Wald chi2(4)    =     42.90
Log likelihood  =   -220.97225                      Prob > chi2     =    0.0000
                              (standard errors adjusted for clustering on id)
------------------------------------------------------------------------------
      _t |               Robust
      _d |      Coef.   Std. Err.       z     P>|z|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  afe_10 |     1.9157   .4655684      4.115   0.000       1.003203    2.828198
  yfe_15 |   .1561891   .3807642      0.410   0.682      -.5900949    .9024732
 yfe2_15 |  -1.430739   .6503867     -2.200   0.028      -2.705474   -.1560048
  logexp |   .7999919   .2140395      3.738   0.000       .3804821    1.219502
------------------------------------------------------------------------------
```

Next, we use the Self and Prentice method.

```
. stcox afe_10 yfe_15 yfe2_15 logexp, offset(_wSelPre) robust nohr nolog
        failure _d:  _d
  analysis time _t:  _t
  enter on or after:  time _t0
                id:  id
Cox regression -- no ties
No. of subjects =          243                     Number of obs   =       260
No. of failures =           56
Time at risk    =  4653.122862
                                                   Wald chi2(4)    =     39.15
Log likelihood  =   -4119.5279                     Prob > chi2     =    0.0000
                              (standard errors adjusted for clustering on id)
------------------------------------------------------------------------------
      _t |               Robust
      _d |      Coef.   Std. Err.       z     P>|z|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  afe_10 |   1.937374   .4830806      4.010   0.000       .9905537    2.884195
  yfe_15 |   .1819405   .4010318      0.454   0.650      -.6040675    .9679484
 yfe2_15 |  -1.477223   .6956013     -2.124   0.034      -2.840576   -.1138694
  logexp |   .8238937   .2278558      3.616   0.000       .3773045    1.270483
_wSelPre |   (offset)
------------------------------------------------------------------------------
```

Now, the Barlow method.

```
. stcox afe_10 yfe_15 yfe2_15 logexp, offset(_wBarlow) robust nohr nolog
        failure _d:  _d
  analysis time _t:  _t
  enter on or after:  time _t0
                id:  id
Cox regression -- no ties
No. of subjects =          243                     Number of obs   =       260
No. of failures =           56
Time at risk    =  4653.122862
                                                   Wald chi2(4)    =     40.38
Log likelihood  =   -286.97533                     Prob > chi2     =    0.0000
                              (standard errors adjusted for clustering on id)
------------------------------------------------------------------------------
      _t |               Robust
      _d |      Coef.   Std. Err.       z     P>|z|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  afe_10 |   1.935024   .4790842      4.039   0.000        .996036    2.874012
  yfe_15 |   .1779033   .3958307      0.449   0.653      -.5979106    .9537172
 yfe2_15 |  -1.475154   .6836251     -2.158   0.031      -2.815035   -.1352738
  logexp |   .8216578   .2236693      3.674   0.000       .3832741    1.260042
_wBarlow |   (offset)
------------------------------------------------------------------------------
```

The first two methods can be implemented using `stselpre` as well. In this command, standard errors derive from the Self and Prentice model-based variance–covariance matrix. Note that they are similar to those calculated using the robust estimator.

```
.stselpre afe_10 yfe_15 yfe2_15 logexp, nohr
        failure _d:  _d
  analysis time _t:  _t
  enter on or after:  time _t0
                id:  id
Method for ties: efron
Self Prentice Variance Estimate for Case-Cohort Design
Self Prentice Scheme
------------------------------------------------------------------------------
        |      Coef.   Std. Err.       z     P>|z|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  afe_10 |   1.937374   .4888502      3.963   0.000       .9792455    2.895503
  yfe_15 |   .1819405   .3806513      0.478   0.633      -.5641223    .9280032
 yfe2_15 |  -1.477223   .6156814     -2.399   0.016      -2.683936   -.2705096
  logexp |   .8238937   .2221188      3.709   0.000       .3885489    1.259239
------------------------------------------------------------------------------
```

```
Prentice Scheme
------------------------------------------------------------------------------
          |      Coef.   Std. Err.       z     P>|z|       [95% Conf. Interval]
----------+-------------------------------------------------------------------
  afe_10 |      1.9157   .4888502      3.919   0.000        .9575717    2.873829
  yfe_15 |    .1561891   .3806513      0.410   0.682       -.5898736    .9022519
 yfe2_15 |   -1.430739   .6156814     -2.324   0.020       -2.637453   -.2240261
  logexp |    .7999919   .2221188      3.602   0.000        .3646471    1.235337
------------------------------------------------------------------------------
```

Saved Results

`stselpre` saves in `e( )` :

Scalars

`e(N)`	number of observations
`e(N_sub)`	number of subjects

Macros

`e(cmd)`	stselpre
`e(ties)`	method for handling ties
`e(predict)`	program used to implement predict
`e(scheme)`	model name of coefficient vector saved

Matrices

`e(b)`	coefficient vector of Prentice method (default). If `self` option is used, coefficient vector of Self and Prentice method are saved.
`e(V)`	Self and Prentice model-based variance–covariance matrix

Functions

`e(sample)`

Methods and formulas

The case–cohort design uses a pseudo-likelihood. Let $z_i(t)$ be the covariate vector and $Y_i(t)$ an at risk indicator for individual i at time t. Then, the pseudo-likelihood can be expressed as a weighted Cox regression model with the weights differently defined according to various analysis methods

$$\frac{Y_i(t)e^{z_i(t_j)\beta}}{Y_i(t)w_ie^{z_i(t_j)\beta}+\sum_{k\in S,\ k\neq i}Y_k(t_j)w_ke^{z_k(t_j)\beta}}$$

In the denominator, contributions of the failure and nonfailures (controls) at risk are given by the first and second terms respectively. In preparing a dataset for case–cohort analysis, all records with a failure are dissected in two time segments: 1) $(t_0, t-\epsilon]$ and 2) $(t-\epsilon, t]$. A case outside the subcohort isn't at risk until just before failure, so the first segment of nonsubcohort cases (and any other previous record of the same observation) is discarded from the sample. Both segments are retained for subcohort members who fail. Thus, the final dataset includes all the observations in the subcohort and just the final time segment records for subjects who fail but aren't in the subcohort. In such a dataset, implementing the Prentice method as in the formula above doesn't require that the weights be specified. In the other two methods, the weights must be specified. In Stata, the weights can be incorporated as an offset term; the logarithm of the weight must be used. The Self and Prentice method employs in the denominator just the subcohort members. This can be accomplished by setting the offset to zero for all observations in the subcohort, whereas for nonsubcohort cases setting the offset to -100, a value corresponding to a weight less than 10^{-40} that effectively allows exclusion of this observation from the denominator of the risk set.

The Barlow method requires that the offset be zero for all records corresponding to failures. Individuals who don't fail and the first time segment of subcohort members who do fail have offset $\log(1/\alpha)$ in an attempt to weight the controls in the denominator to the inverse of the sampling fraction.

Therneau and Li demonstrated that the Self and Prentice variance estimator for case cohort design can be calculated as:

$$V = I^{-1} + (1-\alpha)D'_{SC}D_{SC}$$

where I^{-1} is the usual variance–covariance matrix returned by the Cox model, and D_{SC} is a subset of the matrix of dfbeta residuals that contains only records for the subcohort. (Note that Therneau and Li use a data setup slightly different than the one described in this section.)

Stata can calculate dfbeta residuals using efficient score residuals and I^{-1}. `stdb`, a Stata program at T. Lumley's web site (http://www.biostat.washington.edu/~thomas/), implements this calculus in a way not limited by the `matsize` constraints. When dfbeta residuals are at hand, it is straightforward to obtain the subset of the matrix through the usual `matrix accum` command.

Acknowledgment

The author thanks Dallas English of the Anti-Cancer Council of Victoria, Australia, for checking the commands and revising the insert.

References

Barlow, W. E. 1994. Robust variance estimation for the case-cohort design. *Biometrics* 50: 1064–1072.

Barlow, W. E., L. Ichikawa, D. Rosner, and S. Izumi. 1999. Analysis of case-cohort designs. *Journal of Clinical Epidemiology* 52: 1165–1172.

Breslow, N. E. and N. E. Day. 1987. *Statistical Methods in Cancer Research*, vol. II. Lyon: International Agency for Research on Cancer.

Clayton, D. and M. Hills. 1993. *Statistical Models in Epidemiology*. New York: Oxford University Press.

Langholz, B. and D. C. Thomas. 1990. Nested case-control and case-cohort methods of sampling from cohort: A critical comparison. *American Journal of Epidemiology* 131: 169–176.

——. 1991. Efficiency of cohort sampling designs: Some surprising results. *Biometrics* 47: 1563–1571.

Lin, D. Y. and Z. Ying. 1993. Cox regression with incomplete covariate measurements. *Journal of the American Statistical Association* 88: 1341–1349.

Prentice, R. L. 1986. A case-cohort analysis for epidemiologic cohort studies and disease prevention trial. *Biometrika* 73: 1–11.

Rothman, K. J. and S. Greenland. 1998. *Modern Epidemiology.* 2d ed. Philadelphia: Lippincott Williams & Wilkins.

Self, S. G. and R. L. Prentice. 1988. Asymptotic distribution theory and efficiency results for case-cohort studies. *Annals of Statistics* 16: 64–81.

Therneau, T. M. and Li H. 1999. Computing the Cox model for case cohort designs. *Lifetime Data Analysis* 5: 99–112.

sbe42 | Modeling the process of entry into the first marriage using Hernes model

Duolao Wang, London School of Hygiene and Tropical Medicine, London, UK, duolao.wang@lshtm.ac.uk

Abstract: This article describes the `hernes` command that fits the Hernes model of proportion of a cohort ever-married by age. The Hernes model has been widely applied in demographic studies. The author proposes to use the least squares method for model estimation and illustrates the use of the command with U.S. census data.

Keywords: Hernes model, first marriage, diffusion model.

Hernes (1972) developed a diffusion model for the process of entry into first marriage to explain the bell-shaped hazard rate of entry into marriage. In this model, he posits that two competing structural processes explain the time dependence in the process of entry into marriage. On one hand, with rising age t, there is an increasing proportion $F(t)$ of a cohort that has already entered into first marriage, which in turn enhances the pressure to marry on those who are still unmarried. On the other hand, there is some sort of decreasing social attractiveness and, more importantly, a declining chance $s(t)$ of contact between unmarried peers with increasing time t.

The diffusion model by Hernes can be expressed as

$$\frac{dF(t)}{dt} = s(t)F(t)(1 - F(t)) \tag{1}$$

that is, the rate of change in the proportion married is a function of the proportion already married, the proportion not yet married, and the parameter of conversion that itself is a declining function of time.

We assume simply that each person starts out with a certain marriage potential, A_i, but that this potential declines with a constant proportion b for each time unit, where b is the same for all individuals. For the population as a whole, we then have

$$s(t) = Ab^t \tag{2}$$

where A is the average initial marriageability, and $b < 1$ is the constant of deterioration. Thus, over time, marriageability is a geometric progression; it decreases with a constant proportion for each time unit.

Inserting (2) into (1) we have

$$\frac{dF(t)}{dt} = Ab^tF(t)(1 - F(t)) \tag{3}$$

which can be transposed by integration, resulting in

$$-\log\frac{1 - F(t)}{F(t)}\bigg|_0^t = \frac{Ab^t}{\log b}\bigg|_0^t \tag{4}$$

When $F_0 = F(0)$ is a positive quantity, this expression can be written after exponentiation and rearrangement as

$$F(t) = \frac{1}{1 + \dfrac{(1 - F_0)\exp(A/\log b)}{F_0 \exp(Ab^t/\log b)}} \tag{5}$$

Since A and b are constants, we can always find a number such that

$$\log a = \frac{A}{\log b} \tag{6}$$

If this is substituted into (5), the equation for $F(t)$ becomes

$$F(t) = \frac{1}{\dfrac{1 + (1 - F_0)a}{F_0 a^{b^t}}} \tag{7}$$

which, if we let $k = F_0/(a(1 - F_0))$, reduces to

$$F(t) = \frac{1}{1 + \dfrac{1}{ka^{b^t}}} \tag{8}$$

The curve corresponding to this equation looks somewhat like the logistic, but its inflection point is in general not midway between zero and its upper asymptote, so the limbs of the curve are not symmetric about the inflection point, as the logistic curve is constrained to be.

Note that F_0 is well defined and not equal to zero, but

$$F_0 = \frac{1}{1 + \dfrac{1}{ka}} \tag{9}$$

where k and a are estimated quantities. In practical terms, this means that in an empirical curve fit, we must take as t_0 the first year of the process. Note also that since $b < 1$, we have

$$\lim_{t \to \infty} F(t) = \frac{1}{1 + \dfrac{1}{k}} \tag{10}$$

Thus, the percentage ever married approaches (10) as an asymptote as the cohort ages.

To estimate the model for a given set of observed cumulative marriage rate ($F(t)$), we need a technique to estimate parameters F_0, A, and b in formula (5), or equivalently, k, a, and b in (8) from which F_0 and A can be calculated. Once they are estimated, we can compute the predicted percentage married from the model for increasing values of t and compare these calculated values to the observed ones.

The quantity

$$g(t) = Ab^t \tag{11}$$

in equation (8) has the general form of the so-called Gompertz function, which can be easily estimated.

For that we rewrite (8) as

$$Ab^t = \frac{F(t)}{1 - F(t)} \tag{12}$$

and then use this procedure for the Gompertz function to find estimates for the parameters in equations (5) or (8).

Hernes proposed to use a simple procedure developed by Prescott (1922) for estimating the parameters of the Gompertz curve from the cumulative observations over time. Since the curve has three parameters, three equations are needed to find them. Equation (12) can be rewritten as

$$\log g(t) = \log k + \log ab^t \tag{13}$$

If we can divide the data into three equal sections, Prescott's procedure yields the estimates of the Hernes model from the equations

$$b^T = \frac{\sum_3 \log g(t) - \sum_2 \log g(t)}{\sum_2 \log g(t) - \sum_1 \log g(t)} \tag{14}$$

$$\log a = \left\{ \sum_2 \log g(t) - \sum_1 \log g(t) \right\} \frac{b-1}{(b^T-1)^2} \tag{15}$$

$$\log k = \frac{1}{T} \left\{ \sum_1 \log g(t) - \frac{b^T-1}{b-1} \log a \right\} \tag{16}$$

where $\sum_1$ stands for the sum of logarithms of observed cumulative percentages transformed by formula (12) of the first section, $\sum_2$ the corresponding sum of the second section, $\sum_3$ the sum of the third section, and T the number of observations in each of the three sections.

The above procedure has the advantage of easy calculation but does not provide the measurement of accuracy of estimated parameters. We here propose to use nonlinear least squares to estimate the Hernes model, which can be easily implemented by using Stata's `nl` command.

Syntax

`hernes` *var_age var_rate* [`if` *exp*] [`in` *range*] [`, method(`*string*`)`]

Description

`hernes` generates five parameters for the Hernes model using given age-specific cumulative marriage rates: F_0, A, a, b, and k. In addition, it yields the estimated age-specific cumulated marriage rates and age-specific differences between the observed and estimated cumulative marriage rates.

Options

`method(`*string*`)` specifies the method for estimating the Hernes model, either `hernes` for the Hernes method as described above or `nl` for the nonlinear least squares method. The default is the nonlinear least squares method.

Examples

We use the data of the cumulative first marriages for white women born in 1920–24 in the United States from the U.S. Bureau of Census to demonstrate the use of `hernes` to fit the Hernes model.

```
. use hernes
(Cumulative First Marriages for White Women Born in 1920-24 in the US)
. describe
Contains data from hernes.dta
  obs:            24                          Cumulative First Marriages for
                                                White Women Born in 1920-24 in
                                                the US
 vars:             2                          18 Nov 2000 21:25
 size:           288 (99.9% of memory free)
-------------------------------------------------------------------------------
   1. age       float  %9.0g                  Age in Year
   2. rate      float  %9.3f                  Cumulative First Marriage Rate
-------------------------------------------------------------------------------
Sorted by:
. summ
Variable |     Obs        Mean   Std. Dev.       Min        Max
---------+-----------------------------------------------------
     age |      24        26.5   7.071068         15         38
    rate |      24    .6524583   .3146823       .018       .916
```

We first use the Hernes method:

```
. hernes age rate, method("hernes")
```

```
. list
          age       rate   rate_hat       diff
  1.       15      0.018      0.017      0.001
  2.       16      0.046      0.043      0.003
  3.       17      0.096      0.094      0.002
  4.       18      0.176      0.173      0.003
  5.       19      0.275      0.277     -0.002
  6.       20      0.381      0.391     -0.010
  7.       21      0.483      0.499     -0.016
  8.       22      0.575      0.592     -0.017
  9.       23      0.653      0.667     -0.014
 10.       24      0.716      0.725     -0.009
 11.       25      0.766      0.769     -0.003
 12.       26      0.806      0.803      0.003
 13.       27      0.833      0.829      0.004
 14.       28      0.852      0.849      0.003
 15.       29      0.867      0.864      0.003
 16.       30      0.879      0.876      0.003
 17.       31      0.887      0.886      0.001
 18.       32      0.894      0.893      0.001
 19.       33      0.900      0.900      0.000
 20.       34      0.905      0.905      0.000
 21.       35      0.909      0.909      0.000
 22.       36      0.912      0.912      0.000
 23.       37      0.914      0.915     -0.001
 24.       38      0.916      0.917     -0.001
. return list
scalars:
       r(F0)        =  .9298703505260212
       r(k)         =  13.25930412458492
       r(b)         =  .8544559821043444
       r(a)         =  .001292205436846
       r(A)         =  1.046201407690587
```

and now the `nl` method

```
. hernes age rate, method("nl")
(obs = 24)
Iteration 0:  residual SS =  .0010229
Iteration 1:  residual SS =  .0003782
Iteration 2:  residual SS =  .0003449
Iteration 3:  residual SS =  .0003449
  Source |       SS       df       MS              Number of obs =        24
---------+------------------------------           F(  3,    21) = 253609.83
   Model |   12.494074     3  4.16469133           Prob > F      =    0.0000
Residual |  .000344855    21  .000016422           R-squared     =    1.0000
---------+------------------------------           Adj R-squared =    1.0000
   Total |  12.4944188    24  .520600785           Root MSE      =  .0040524
                                                  Res. dev.     = -199.5015
(hernes)
------------------------------------------------------------------------------
    rate |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
       a |   .0013287   .0000492     27.019   0.000       .0012264     .001431
       b |   .8624038   .0018716    460.795   0.000       .8585117    .8662959
       k |   14.53891   .4107641     35.395   0.000       13.68468    15.39314
------------------------------------------------------------------------------
 (SE's, P values, CI's, and correlations are asymptotic approximations)
. list
          age       rate   rate_hat       diff
  1.       15      0.018      0.019     -0.001
  2.       16      0.046      0.046      0.000
  3.       17      0.096      0.095      0.001
  4.       18      0.176      0.172      0.004
  5.       19      0.275      0.272      0.003
  6.       20      0.381      0.382     -0.001
  7.       21      0.483      0.488     -0.005
  8.       22      0.575      0.581     -0.006
  9.       23      0.653      0.657     -0.004
 10.       24      0.716      0.717     -0.001
 11.       25      0.766      0.763      0.003
```

```
 12.        26      0.806      0.798      0.008
 13.        27      0.833      0.826      0.007
 14.        28      0.852      0.847      0.005
 15.        29      0.867      0.863      0.004
 16.        30      0.879      0.876      0.003
 17.        31      0.887      0.887      0.000
 18.        32      0.894      0.895     -0.001
 19.        33      0.900      0.902     -0.002
 20.        34      0.905      0.907     -0.002
 21.        35      0.909      0.912     -0.003
 22.        36      0.912      0.915     -0.003
 23.        37      0.914      0.918     -0.004
 24.        38      0.916      0.921     -0.005
```

As the above results show, both Hernes and least squares estimates fit the data very well, and the latter method yields a much better fit than the former. The largest deviations from the least squares procedure are found for ages 26 and 27 at which points the percentages married are overestimated by 0.8% and 0.7%, respectively. For the Hernes method, the largest difference is found for age 22 at which point the percentage married is overestimated by 1.72%. The degree of fit for the least squares method is seen more clearly in Figure 1, which plots the observed cumulative first marriages against age together with the fitted curve.

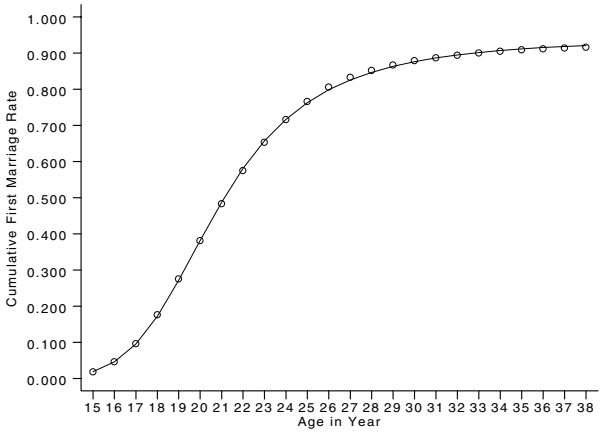

Figure 1. Observed and fitted first marriages versus age using nonlinear least squares.

The `hernes` command can also be applied conditionally using `if` and `in` expressions as shown by

```
. hernes age rate if age<=24
(obs = 10)
Iteration 0:  residual SS =  .0203066
Iteration 1:  residual SS =  .0061118
Iteration 2:  residual SS =  .0052664
Iteration 3:  residual SS =  .0043897
Iteration 4:  residual SS =  .0034274
Iteration 5:  residual SS =  .0029379
Iteration 6:  residual SS =   .001443
Iteration 7:  residual SS =  .0000468
Iteration 8:  residual SS =   .000045
Iteration 9:  residual SS =   .000045
  Source |       SS       df       MS              Number of obs =        10
---------+------------------------------           F(  3,     7) =  91576.69
   Model |  1.76635203     3  .588784009           Prob > F      =    0.0000
Residual |  .000045006     7  6.4294e-06           R-squared     =    1.0000
---------+------------------------------           Adj R-squared =    1.0000
   Total |  1.76639703    10  .176639703           Root MSE      =  .0025356
                                                  Res. dev.     = -94.73426

(hernes)
------------------------------------------------------------------------------
    rate |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
       a |   .0014192   .0000848     16.740   0.000       .0012187    .0016197
       b |     .86433   .0039088    221.123   0.000        .855087    .8735729
       k |   14.52711   1.341251     10.831   0.000       11.35555    17.69867
------------------------------------------------------------------------------
 (SE's, P values, CI's, and correlations are asymptotic approximations)
```

Saved results

`hernes` saves the following scalars in `r()`:

`r(F0)`	F_0
`r(A)`	A
`r(a)`	a
`r(b)`	b
`r(k)`	k

where these quantities are the parameters in equations (7) and (8).

References

Hernes, G. 1972. The Process of entry into first marriage. *American Sociological Review* 37: 173–182.

Prescott, R. B. 1922. Law of growth in forecasting demand. *Journal of the American Statistical Association* 18: 471–479.

sg97.3 | Update to formatting regression output

John Luke Gallup, developIT.org, jgallup@maine.rr.com

Abstract: An update to the `outreg` command is described.

Keywords: regression output.

I have fixed some small bugs in `outreg`, a program described in Gallup (1998, 1999, 2000) that writes regression output to a text file.

References

Gallup, J. L. 1998. sg97: Formatting regression output for published tables. *Stata Technical Bulletin* 46: 28–30. Reprinted in *Stata Technical Bulletin Reprints*, vol. 8, pp. 200–202.

——. 1999. sg97.1: Revision of outreg. *Stata Technical Bulletin* 49: 23. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, pp. 170–171.

——. 2000. sg97.2: Revision of outreg. *Stata Technical Bulletin* 58: 9–13.

sg158 | Random-effects ordered probit

Guillaume R. Frechette, Ohio State University, gurst1@econ.ohio-state.edu

Abstract: The program `reoprob` that estimates a random-effects ordered probit model is presented and shown to be significantly faster than `gllamm6`. This increase in speed stems from the use of analytical first derivatives in the computation of the quasi-newton step.

Keywords: random-effects ordered probit, `gllamm6`, quasi-Newton algorithm.

Introduction

Recent developments in computing power have allowed the estimation of increasingly complex problems. One such class of estimators is allowing for individual specific effects when analyzing limited dependent variables. The first example of this in Stata is `rfprobit` introduced by Sribney (1995). This was followed by the inclusion of the random-effects option in `xtprobit`, and more recently by the creation of `gllamm6` by Rabe-Hesketh, et al. (2000), which allows the computation of random-effects ordered probit models. However, the latter relies solely on the computation of the likelihood for the optimization, that is, the first derivatives and the Hessian are numerically approximated, and thus can be very slow, even for relatively simple problems. I propose a program, `reoprob`, which makes use of the analytical first derivatives and, thus, considerably improves performance. Using `gllamm6` as a benchmark, I will show that this new program finds the "correct" solution and that it is substantially faster.

Syntax

`reoprob` *depvar varlist* [`if` *exp*] [`in` *range*] , `i(`*varname*`)` [`quadrat(#)` `level(#)` *maximize_options*]

This command shares the features of all estimation commands. `reoprob` typed without arguments redisplays previous results.

Options

`i(`*varname*`)` is not optional, it specifies the variable corresponding to an independent unit (for example, a subject id).

`quadrat(#)` specifies the number of points to use for Gaussian–Hermite quadrature. It is optional, and the default is 12. Increasing this value improves accuracy, but also increases computation time. Computation time is roughly proportional to its value.

`level(#)` specifies the confidence level, in percent, for confidence intervals. The default is `level(95)` or as set by `set level`.

maximize_options controls the maximization process and the display of information; see [R] **maximize**. `nolog` suppresses the display of the likelihood iterations. Use the `trace` option to view parameter convergence. The `ltol(#)` and `tol(#)` options can be used to loosen the convergence criterion (respectively 1e-7 and 1e-6 by default) during specification searches. `iter(#)` specifies the maximum number of iterations.

Remarks

The problem of interest can be describe as wanting to estimate

$$\begin{aligned} y_{it}^* &= \beta' x_{it} + \epsilon_{it}, \qquad i = 1, \ldots, N,\ t = 1, \ldots, T \\ \epsilon_{it} &= v_{it} + u_i \\ \text{Var}(\epsilon_{it}) &= \sigma_v^2 + \sigma_u^2 = 1 + \sigma_u^2 \\ \text{Corr}(\epsilon_{it}, \epsilon_{is}) &= \rho = \frac{\sigma_u^2}{1 + \sigma_u^2} \end{aligned}$$

where y^* is unobserved. Instead, the analyst observes

$$y_{it} = \begin{cases} 0 & \text{if } y_{it}^* \leq \mu_0, \\ 1 & \text{if } \mu_0 < y_{it}^* \leq \mu_1 \\ 2 & \text{if } \mu_1 < y_{it}^* \leq \mu_2 \\ \vdots & \\ \text{J} & \text{if } \mu_{J-1} < y_{it}^* \end{cases}$$

Define $a_{it} = \mu_{j-1} - \beta' x_{it}$ and $b_{it} = \mu_j - \beta' x_{it}$ if $y_{it} = j$, where $\mu_{-1} = -\infty$ and $\mu_J = \infty$. Then, the log-likelihood function is $L = \sum_{i=1}^{N} \ln\left(P\left(y_{i1}, y_{i2}, \ldots, y_{iT}\right)\right)$ where, by simply generalizing the argument made in Butler and Moffitt (1982), one can show that

$$\begin{aligned} P(y_{i1}, y_{i2}, \ldots, y_{iT}) &= \int_{a_{i1}}^{b_{i1}} \cdots \int_{a_{iT}}^{b_{iT}} f(\epsilon_{i1}, \ldots, \epsilon_{iT}) d\epsilon_{iT} \cdots d\epsilon_{i1} \\ &= \int_{a_{i1}}^{b_{i1}} \cdots \int_{a_{iT}}^{b_{iT}} \int_{-\infty}^{\infty} \prod_{t=1}^{T} f\left(v_{it}|u_i\right) f\left(u_i\right) du_i dv_{iT} \cdots dv_{i1} \\ &= \int_{-\infty}^{\infty} \prod_{t=1}^{T} \left[F(b_{it}|u_i) - F(a_{it}|u_i)\right] du_i \end{aligned}$$

in which $f(\cdot)$ and $F(\cdot)$ represent the pdf and cdf of the normal distribution function, respectively. As Butler and Moffitt (1982) demonstrated, this is amenable to Gaussian quadrature. Of course, this is sufficient to estimate such a model as one can use numerical approximation to the first and second derivatives to compute quasi-Newton steps. This, however, makes every step fairly long to compute, even for a relatively small sample. This can be improved upon since the first derivatives can also be approximated by Gauss–Hermite quadrature. Using the convention that $f_{it}^j = f\left(\mu_j - \beta' x_{it}\right)$, $F_{it}^j = F\left(\mu_j - \beta' x_{it}\right)$, $L_i = P\left(y_{i1}, y_{i2}, \ldots, y_{iT}\right)$, and an indicator function $1\left\{statement\right\}$ which takes value 1 if the statement is true and 0 otherwise, the first derivative with respect to a parameter k is given by

$$\frac{\partial L}{\partial k} = \frac{\partial \sum_i \ln L_i}{\partial k} = \sum_i \frac{1}{L_i} \frac{\partial L_i}{\partial k}$$

and thus for our parameters of interest

$$\frac{\partial L_i}{\partial \beta} = \int_{-\infty}^{\infty} f(u_i) \sum_{t=1}^{T} \frac{f_{it}^{j-1} - f_{it}^{j}}{F_{it}^{j} - F_{it}^{j-1}} x_{it} \prod_{t=1}^{T} \left[F(b_{it}|u_i) - F(a_{it}|u_i)\right] du_i$$

$$\frac{\partial L_i}{\partial \mu_j} = \int_{-\infty}^{\infty} f(u_i) \sum_{t=1}^{T} \frac{f_{it}^{j-1} 1\{y_{it} = j\} - f_{it}^{j} 1\{y_{it} = j-1\}}{F_{it}^{j} - F_{it}^{j-1}} \prod_{t=1}^{T} \left[F(b_{it}|u_i) - F(a_{it}|u_i)\right] du_i$$

$$\frac{\partial L_i}{\partial \rho} = \int_{-\infty}^{\infty} f(u_i) \sum_{t=1}^{T} \frac{f_{it}^{j-1} - f_{it}^{j}}{F_{it}^{j} - F_{it}^{j-1}} \frac{\sqrt{1-\rho}}{\sqrt{2\rho}(1-\rho)} \prod_{t=1}^{T} \left[F(b_{it}|u_i) - F(a_{it}|u_i)\right] du_i$$

Taking advantage of knowing the analytical first derivatives will result in substantial speed improvements, as will be shown below.

Examples

To demonstrate and test `reoprob`, I will investigate the effect of income, schooling, and political freedom on the degree of bureaucratic corruption in non-OECD countries. To this end, I will use data from 87 non-OECD countries over a 16 year period (1982–1997) on the level of bureaucratic corruption produced by the International Country Risk Guide. Not all years are available for every country, however. The corruption index (CI) ranges from 0 to 6. It is reported on a monthly basis, but I am using annual averages. Lower scores indicate "high government officials are likely to demand special payments" and that "illegal payments are generally expected throughout lower levels of government" in the form of "bribes connected with import and export licenses, exchange controls, tax assessment, police protection, or loans." Although the index does take noninteger values (because of the averaging), 90% of the values are integers and thus the rest are recoded to the smallest integer. Income and schooling are from the Global Development Network Growth Database developed by the World Bank. Income is taken to be GDP per capita, and education is measured as the ratio of total enrollment in primary school, regardless of age, to the population of the age group that officially corresponds to the primary school level. Estimates are based on the International Standard Classification of Education. Political freedom (PF) is given by the Gastil index of political rights. The Gastil index ranges from 1 to 7, one being the highest degree of political freedom.

For the purpose of comparison, however, I will first look at a simplified problem. First, I will reduce the number of values for the dependent variable to 3, that is, CI3 is 0 if CI is 0 or 1, it is 1 if CI is 2 or 3, and is 3 if CI is 4 or 5. The only regressor will be income. This will be estimated using 12 points for the quadrature. Using a Pentium III 450 mhz with 256 megs of RAM on a Windows NT 4.0 workstation platform, `reoprob` took 2 minutes 37 seconds to converge to the result presented below.

```
. use stb
. describe
Contains data from stb.dta
  obs:         1,068
 vars:             6                          27 Nov 2000 14:04
 size:        29,904 (97.1% of memory free)
-------------------------------------------------------------------------------
   1. y          float  %9.0g
   2. ps         float  %9.0g
   3. pol        float  %9.0g
   4. corr6      float  %9.0g
   5. corr3      float  %9.0g
   6. id         float  %9.0g                 group(id)
-------------------------------------------------------------------------------
Sorted by:  id
. reoprob corr3 y, i(id) quad(12)
Fitting constant-only model:
Iteration 0:   log likelihood = -677.80343
Iteration 1:   log likelihood = -600.94473
Iteration 2:   log likelihood = -591.25914  (not concave)
Iteration 3:   log likelihood = -577.09022
Iteration 4:   log likelihood =  -561.8791
Iteration 5:   log likelihood = -560.88178
Iteration 6:   log likelihood = -560.88048
Iteration 7:   log likelihood = -560.88048
Fitting full model:
Iteration 0:   log likelihood = -669.98703
```

```
Iteration 1:   log likelihood = -607.63587  (not concave)
Iteration 2:   log likelihood = -587.38208  (not concave)
Iteration 3:   log likelihood = -574.74158
Iteration 4:   log likelihood = -560.16828
Iteration 5:   log likelihood = -559.90922
Iteration 6:   log likelihood = -559.90892
Iteration 7:   log likelihood = -559.90892
Random Effects Ordered Probit                     Number of obs   =       1068
                                                  LR chi2(1)      =       1.94
Log likelihood = -559.90892                       Prob > chi2     =     0.1633
------------------------------------------------------------------------------
       corr3 |      Coef.   Std. Err.       z     P>|z|       [95% Conf. Interval]
-------------+----------------------------------------------------------------
eq1          |
           y |  -.0308618   .0202292     -1.526   0.127      -.0705104    .0087868
-------------+----------------------------------------------------------------
_cut1        |
       _cons |  -1.066998   .1307553     -8.160   0.000      -1.323274   -.8107222
-------------+----------------------------------------------------------------
_cut2        |
       _cons |   2.422023   .1522812     15.905   0.000       2.123557    2.720488
-------------+----------------------------------------------------------------
rho          |
       _cons |   .7618085   .0238483     31.944   0.000       .7150667    .8085504
------------------------------------------------------------------------------
```

Using `gllamm6` proves to be substantially slower; it converges to exactly the same coefficient estimates in 8 minutes 43 seconds, over 3 times slower.

Estimating the complete model yields similar results. Regressing CI on income, education, and PF, it takes 13 minutes and 14 seconds for `reoprob` to converge versus 44 minutes 35 seconds for `gllamm6`; again more than three times slower. Moreover, in this specific case, `reoprob` stops at a "better" coefficient estimate. This, however, should not be expected to be true in general. There may well be problems for which the opposite is true, it is simply a question of the stopping criterion being affected by the differences in the analytical and numerical gradients. These results are presented in the table of determinants of corruption in non-OECD countries given below.

	`reoprob`	`gllamm6`
income	−0.074	−0.034
	(0.014)***	(0.017)*
education	−0.006	−0.002
	(0.002)***	(0.002)
PF	0.134	0.198
	(0.032)***	(0.029)***
cut1	−3.360	−1.694
	(0.239)***	(0.278)***
cut2	−1.728	−0.025
	(0.228)***	(0.273)
cut3	0.341	2.073
	(0.222)	(0.272)***
cut4	1.876	3.637
	(0.221)***	(0.283)***
cut5	3.004	4.754
	(0.241)***	(0.308)***
rho	0.755	
	(0.015)***	
var(1)		1.980
		(0.170)***
Observations	1068	1068
Log Likelihood	−1087.556	−1091.551

Also worth observing is that even though the two programs have stopped at a slightly different point, all coefficient estimates are of the same sign.

Hence, this paper has shown that `reoprob` computes the likelihood for a random-effects probit correctly. Furthermore, it has provided examples of the considerable increase in speed that may be achieved.

References

Butler, J. S. and R. Moffitt. 1982. A computationally efficient quadrature procedure for the one-factor multinomial probit model. *Econometrica* 50: 761–764.

Greene, W. H. 2000. *Econometric Analysis*. Upper Saddle River, NJ: Prentice Hall.

Rabe-Hesketh, S., A. Pickles, and C. Taylor. 2000. sg129: Generalized linear latent and mixed models. *Stata Technical Bulletin* 53: 47–57. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, pp. 293–307.

Sribney, W. 1995. sg41: Random-effects probit. *Stata Technical Bulletin* 26: 15–18. Reprinted in *Stata Technical Bulletin Reprints*, vol. 5, pp. 107–111.

sg159	Confidence intervals for correlations

Paul T. Seed, King's College London, UK, paul.seed@kcl.ac.uk

Abstract: The commands `ci2` and `cii2` are introduced and illustrated. They perform as Stata's `ci` and `cii`, but also give confidence intervals for Pearson's product moment correlation and Spearman rank correlation, based on Fisher's transformation.

Keywords: correlation coefficient, Fisher's transformation, Spearman's rank correlation.

Stata's `ci` command and its immediate version `cii` together provide confidence intervals for a number of statistics based on standard distributions. These include means (normal distribution), proportions (binomial), and expected frequencies (poisson).

Syntax of ci and cii

`ci` [*varlist*] [*weight*] [`if` *exp*] [`in` *range*] [`,` `level(#)` `binomial` `poisson` `exposure(`*varname*`)`
`by(`*varlist*$_2$`)` `total`]

`cii` #obs #mean #sd [`,` `level(#)`] (normal variable)

`cii` #obs #succ [`,` `level(#)`] (binomial variable)

`cii` #exposure #events [`,` `poisson` `level(#)`] (Poisson variable)

`aweights` and `fweights` are allowed; see [U] **14.1.6 weight.**

Brief review of correlation confidence intervals

I will give a reminder of the algebra of correlations, following Altman (1991). For paired data, (x_i, y_i), Pearson's r, the product-moment correlation coefficient, is calculated as

$$r = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}$$

The sampling distribution of Pearson's r is not normal, but we can transform r to get a quantity called z, which does have an asymptotic normal sampling distribution. This is known as Fisher's transformation

$$z = \frac{1}{2} \log \left(\frac{1+r}{1-r} \right)$$

The standard error of z is approximately $s_z = 1/\sqrt{n-3}$, where n is the sample size, so for example, we can construct a 95% probability interval for z as $z_l = z - 1.96 s_z$ to $z_u = z + 1.96 s_z$, which when we transform back via $r = (e^{2z} - 1)/(e^{2z} + 1)$ gives a 95% confidence interval for the population correlation coefficient as $(e^{2z_l} - 1)/(e^{2z_l} + 1)$ to $(e^{2z_u} - 1)/(e^{2z_u} + 1)$.

For nonnormal data, Spearman's rank correlation can be used. For n larger than about 10, Fisher's transformation again gives an approximately normal sampling distribution and a suitable confidence interval.

The new commands `ci2` and `cii2` behave exactly as `ci` and `cii`, except for the extra options `corr` and `spearman`. When these are used, confidence intervals for Pearson's product moment or Spearman's rank correlations are produced.

Syntax for ci2 and cii2

`ci2` *var1 var2* [*weight*] [`if` *exp*] [`in` *range*] [`,` `level(#)` {`corr` | `spearman`}]

`cii2` $\#_n$ $\#_r$ [`,` `level(#)` `corr`]

It is important to include the `corr` and `spearman` options. Without them, these formats are invalid. `ci2` (as `ci`) will give confidence intervals for the mean of the two variables; `cii2` (as `cii`) will give a confidence interval for a binomial distribution, correcting $\#_r$ to give the proportion corresponding to nearest whole number of events.

Examples

We begin with `cii2` operating as `cii`, and as for correlations.

```
. cii2 18 .7921
                                                    -- Binomial Exact --
Variable |     Obs        Mean    Std. Err.       [95% Conf. Interval]
---------+--------------------------------------------------------------
         |      18     .7777778    .0979908        .5236138    .9359131
. cii2 18 .7921, corr
Confidence interval for correlation,
based on Fisher's transformation.
Correlation = 0.792 on 18 observations (95% CI: 0.516 to 0.919)
```

Next, we use `ci2` for `weight` and `mpg` for Stata's auto data using both the `corr` and `spearman` options.

```
. use auto
(1978 Automobile Data)
. ci2 weight mpg, corr
Confidence interval for Pearson's product-moment correlation
of weight and mpg, based on Fisher's transformation.
Correlation = -0.807 on 74 observations (95% CI: -0.874 to -0.710)
. ci2 weight mpg, spearman
Confidence interval for Spearman's rank correlation
of weight and mpg, based on Fisher's transformation.
Correlation = -0.858 on 74 observations (95% CI: -0.908 to -0.782)
```

Finally, we use `spearman` on the first 10 observations.

```
. ci2 weight mpg in 1/10, spearman
Confidence interval for Spearman's rank correlation
of weight and mpg, based on Fisher's transformation.
Correlation = -0.606 on 10 observations (95% CI: -0.894 to 0.039)
Warning: This method may not give valid results
with small samples (n<= 10) for rank correlations.
```

Saved results

Whichever command is used for correlations, the following are saved in `r( )`:

`r(n)`	number of observations
`r(r)`	correlation
`r(lb)`	lower bound
`r(ub)`	upper bound
`r(corr)`	(`ci2` only) correlation type

Otherwise, results are saved as for `ci` and `cii`.

Reference

Altman D. G. 1991. Relationship between two continuous variables. In *Practical Statistics for Medical Research*, ed. D. G. Altman, 277–321. London: Chapman and Hall.

ssa14	Global and multiple causes-of-death life tables from complete or aggregated vital data

Carlos Ramalheira, Coimbra Faculty of Medicine & University Hospital, Coimbra, Portugal, cramal@ci.uc.pt

Abstract: The command `lifetabl` for performing a wide variety of life table analyses is introduced and illustrated.

Keywords: life table analysis, vital statistics, causes of death, survival analysis.

The life table is one of the fundamental tools of vital statistics analysis, either when used from an epidemiological standpoint or from the perspective of actuarial science. As we know it today, the method was formally established in the transition of the 17th to the 18th century by Edmund Halley (1693) and John Graunt (1665), and afterwards became a focus of attention to many other distinguished men of science, such as Benjamin Gompertz (1825), in the first quarter of the 19th century. Although interest in mortality records can be traced back to the 3rd century Roman Empire mortality registries, the systematic compilation and publication of mortality statistics only began by the end of the 19th century. For instance, the first American official life table ever published came to light in 1900 (Selvin 1991).

Life table construction and analysis provides an alternative to standardization as an appropriate method to describe the pattern of the survival experience of a large population or of one of its subgroups, given only that we possess a set of age-specific mortality rates, or more elementary data allowing their computation, such as number of deaths, and midyear population estimates for each age strata (Armitage and Berry 1994; Chiang 1984). Some developments of this approach are also useful to evaluate the impact of competing risks as they act upon a group, as well as to obtain data to draw survival curves, survival probabilities, and hazard functions (Selvin 1991).

Generally, a distinction is made between cohort (or generation) life tables and current life tables. While the former variant aims at describing the actual observed survival experience of a group or cohort of individuals born at about the same time (a generation cohort) followed up through time, the latter type describes the survival pattern of a population group subject throughout life to the age-specific death rates currently observed in a particular community, as though no significant cohort effects, for example, generation variability influences exert their actions.

Both these two life table forms are quite useful in the context of epidemiological or vital statistics studies. While the current life table technique provides an alternative method to standardization when comparing the mortality experience or the burden of disease of different groups, the generation life table approach is particularly useful in the context of occupational health studies; namely, to investigate the patterns of observed mortality in specific professional groups followed up over a long period of time (Armitage and Berry 1994).

Another commonly-made distinction separates abridged from complete life tables. While an abridged life table typically displays data by 5- or 10-year age intervals, a complete life table exhibits data for every single year of age until the closing category which, until recently, was most of times 85+, that is, 85 and over years of age. In this sense, the abridged life table is constructed by reference to a "standard" complete table, whose significant internal relationships are transposed to the simplified version. Being an approximation, its use is mainly justified by computational constraints, or by lack or scarcity of the data. On the other hand, a complete life table may also be aggregated into 5- or 10-year age groups (Anderson 1999).

Chiang (1984), as well as others (for example, Armitage and Berry 1994; Anderson 1999) emphasize that the technique of construction of life tables, such as those published by life assurance offices or national sources of vital statistics, is a rather complex one. However, Hill and Hill (1991), as well as Selvin (1991) describe quite simple construction strategies which, being simplified and accessible to almost anyone, may prove quite useful either for pedagogic purposes, for epidemiological research, or for surveillance.

Syntax

```
lifetabl varlist [if exp] [in range] , strata(age_level_var) [ rates(ratesvar) deaths(deathsvar)
    pop(popvar) by(byvar) nyears(age_interval_length_var) radix(#) sclist(varlist) allrx
    weights(#1[#2[#3[#4]]]) pyll(#) keep label(labelvar) multiplier(#) not noo allsct
    sconly noyll saving(filename) replace grphs ge gp gs gh gsc lograte graph_options ]
```

Description

The `lifetabl` command allows the computation of life tables for whole populations, or their subgroups, based on vital statistics data, following the method described by Selvin (1991). Furthermore, it also permits the user to produce the so-called "multiple specific causes of death analysis," as well as several indicators of potential years of life lost due to specific causes of death, and also a number of graphics describing several aspects of the global and cause-specific survival experience of the groups considered.

`lifetabl` allows the construction of life tables based on data for single years of age, or aggregated for several years, following the general method provided by Selvin. According to this approach, a population life table, as well as several derived statistics, such as the at-birth, e_0, and age-specific, e_j, expectation of life estimates or the corresponding survival probabilities or hazard rates (and functions), can be generated if we dispose of a set of observed age-specific mortality rates (or data allowing their computation, such as the number of recorded deaths and number of persons at risk for each age interval). The command options provided allow not only the computation of general life tables (for all causes of death at once, that is, single-cause life tables), according to desired `if` or `in` clauses, but also for specific population strata, for example, sex groups discriminated by means of a `by` option.

In addition to the tables just mentioned, `lifetabl` also allows the analysis of the impact of multiple specific causes of death (option `sclist`) on the pattern of human mortality. Up to 20 different variables registering the number of deaths related to particular causes in each age strata may be added through the `sc` option. The resulting output will then include tables with absolute values (and rates) for a number of indicators of potential years of life lost connected to the causes of death considered: PYLL$(x)_i$, which is Potential Years of Life Lost until age x due to cause i, PYLL$(x)u_i$, "uncorrected" Potential Years of Life Lost until age x, due to cause i; PEYLL, Period Expected Years of Life Lost; SEYLL, Standard Expected Years of Life Lost (Murray and Lopez 1996), together with tables displaying the following statistics: at birth, and lifetime (beyond age x), conditional probabilities of death for each specific cause (P_{xi}, the probability of death by cause i given survival to age x), absolute risk of dying by each cause, given a certain age, in the following interval (q_{xi}); the cumulative distributions of deaths by cause and age (F_{xi}); and the probabilities of death after a certain age x, given that the death is caused by each of the specific categories considered (S_{xi}).

Moreover, if requested by the option `allsct`, the program will also display tables for observed number of deaths by each cause and age level (Dxi), for the life table expected number of deaths, by each cause at each age level x (dxi), and, finally, for the life table expected total number of deaths by each cause occurring after each age x (Wxi).

Throughout the program output, the notation used for the labeling of life table columns, as well as indicators such as those just described (letters and expressions between parenthesis above), strictly follows the conventions adopted by Selvin (1991). On the other hand, the output labeling for statistics related with the potential years of life lost group of indicators derives from the suggestions of Murray and Lopez (1996). See the *Methods and formulas* section below for further details on the procedures and definitions.

With the `lifetabl` command, it is also possible to produce five different groups of graphics. Options `ge`, `gp`, `gs`, `gh`, and `gsc` allow users to request the graphs of, respectively, expectation of life function(s), cumulative distribution(s) of expected deaths, survival function(s), hazard rate function(s), as well as other graphics related to the specific causes of death included in the command. Option `grphs` requests all five graphics at once.

Since the program has several options to control the extent and type of outputs (for example, `not`, `noo`, `noyll`, `allsct`) and can produce very long series of tables or a large number of graphics, particularly when the `by` or `grphs` options are used together, we strongly recommend the user to begin by exploring the program with one of the example databases included with this insert.

Minimal specification

In order to produce a life table with the `lifetabl` routine, it is necessary to specify at least

1. Through the command option `strata` a numeric or string variable, coded in such a way as to adequately describe the different age strata (for example `1, 2, 3, ... j` if the strata variable is numeric, or `"[00-01[","[01-02[",... "j"` if the variable is string type),

   and one of

2a. A variable containing observed strata specific mortality rates (option `rates`)

2b. Two variables coded with, respectively, the number of death events registered for each age during the time interval considered (option `deaths`), and the number of persons in the same age levels (usually the mid-period population estimates; option `pop`).

Regarding the `strata` option, the values used to designate the successive age intervals must be unique within `if` or `in` subgroups and must also keep the natural order of the successive levels when sorted by the program. For example, a string variable coded `"[0-1[", "[1-2[", "[2-3[",...,"[30-31[", "[31-32[",` and so on, would not conform to this requirement since the third level, `"[2-3["` would always appear after `"[19-20["` after a sorting. When in doubt, perhaps a safer alternative is provided by the use of a naturally ordered numeric variable to designate the successive strata in this option, together with a string variable, which may be declared through option `label` to provide labels for the output. This alternative instructs the program to use as labels in the output the first seven characters of the strings contained in *labelvar*.

If the program is run through the specification of observed rates (`rates(`*ratesvar*`)`), these must be constructed in order to represent number of events for 1, 10, 100, 1,000, 10,000, 100,000, or 1,000,000 persons. If the rates multiplier is other than 100,000 (the default value), the exact value to which the recorded rates are referred to must be explicitly declared through the option `multip(#)`. Use of this last option is allowed only when option `rates(`*ratesvar*`)` is also utilized. However, regardless of the power of ten of the rates used, the figures shown in the life table column `Rx` for the life-table age-specific mortality rates are always referred to number of events per 100,000 individuals.

By the use of option `radix(#)`, where radix stands for l_0, the arbitrary number for people entering the life table's first age level, it is also possible to choose a value different from the default of 100,000. Although the radix is an essential element from which all the life table is derived, this option rarely will need to be used, since the default will be adequate for the majority of circumstances.

Another important technical detail of the life table construction regards the evaluation of the contribution made to the total time lived by a cohort who entered any age interval $(l_0, l_1, \ldots, l_x)$ by those who die in that same period $(d_0, d_1, \ldots, d_x)$. Generally, it is assumed that, on average, those who die during each age period were alive for a time of approximately half of the total interval length. However, for life tables in which the interval lengths considered are equal to one year for ages following birth (complete life tables), that approximation is not valid. Indeed, during the first few years of life, deaths tend to occur not symmetrically around the midpoint of the interval, thus originating an overestimation of the same interval life table stationary population (L_x). Usually this problem is dealt with by some form of correction made to the general rule above. One of the methods available, reported, and adopted by Selvin (1991) is the specification of weights other than 0.5 for the contribution of the death cases to the total time lived in the interval in the first four years of life. The `lifetabl` option `weights` allows the specification of such weights. If the option is ignored and the interval lengths of the life table intervals are all of one year, the weights automatically used by the program will be those empirically determined by Chiang (1984) and reported by Selvin (1991) for, respectively, the first four years of life: 0.113, 0.430, 0.450, and 0.470. At all subsequent intervals and in all circumstances, the contribution of each death to total time lived in the age segment is considered to be equivalent to one half the period length (that is, 0.5 years for complete life tables or half the interval length for tables with different aggregation of ages). On the other hand, if any of the life table age intervals extends itself for more than one year, the program automatically makes this parameter equal to 0.5 at all age levels, unless otherwise explicitly required by the declaration of a different set of one to four values for the first four periods through option `weights`. One other option, `nyears(`*numvar*`)`, allows the user to inform the program about the name of the numeric variable which registers, in years, the length of each age interval. If this option is not used, the program assumes that all age periods are of one year length.

It is also possible to retain and/or save for further analysis or transformation several variables produced in the background during the construction of the life table (options `keep` and `saving(`*filename*`)`). Unless one of the two options just mentioned is included in the command line, the original data will always be restored (or reconstructed) at the end of the processing, thus protecting any user variables with names similar to the ones listed below in the original file. Kept variable names and their content are the following: `_Rx` (life table mortality rates per 100,000 persons), `_qx` (probability of dying in the x age interval), `_dx` (expected number of deaths in the age interval), `_px` (probability of surviving the age interval), `_lx` (number of people alive at the beginning of the age interval), `_Lx` (cumulative years lived through age x), `_Tx` (total time lived beyond age x), `_ExpYL` (expected years of life at age x), `_Surv` (survival probability), `_SurvVar` (Greenwood variance for `_Surv`), and `_Hrate` (hazard rate).

If specific causes of death are also being considered in the analysis requested at the command line, three other sets of variables, respectively suffixed `1, 2, ... i` (the number of the last specific cause of death included in the `sc` option), will also be retained: `_qxi` (risks of dying of cause i in the age interval x), `_Wxi` (expected total number of deaths by cause i after the beginning of the age interval x), and `_Fxi` (cumulative distribution of deaths by cause i, through age).

Other available resources

Stata already has several quite extended and useful commands, the set of `st` algorithms, which can be applied to the analysis of "cohort" type, follow-up data. With some of these `st` commands, it is also possible to produce several variants of the classic life table here presented, provided the original data are recorded at the individual level (single observations followed-up through fixed or variable length time intervals). However, with the available `st` commands, it is not possible to produce life tables for whole populations based on vital statistics data aggregated at the level of age groups as shown in the examples provided below.

Options

`strata(`*age_level_var*`)` allows the specification of a numeric or string variable coding the different age strata. This option must always be specified. If the variable used is numeric, the values representing the successive age strata must be monotonically ascending numbers, because during processing, this variable will be subject to an ascending sort, and the concrete values recorded will be used as labels for the strata (for example, `0, 1, 2, ..., 90` for, respectively, the intervals `[0-1[`, `[1-2[`, `[2-3[`, `... [90+[`). If the alternative possibility of specifying a string variable is utilized, the same also applies.

Therefore, care must be taken in order to use strings which, after being sorted, keep in the correct order the corresponding age strata. A perhaps safer alternative is the use of a numeric variable with ascending integers to designate the successive age levels in this option together with a string variable to provide labels for the different age strata in the output through option `label`. In every circumstance, the (within `if`, `in`, or `by` subgroups) variable describing the age strata cannot have repeated, neither missing nor null, values.

`rates(`*ratesvar*`)` designates a variable for stratum-specific observed mortality rates. Use of this option is mandatory, unless the following two options are used instead. The age-specific rates can be expressed in terms of several population multipliers (powers of ten). However, if the specific power of 10 for which the rates are referred is not 100,000 (the default), its value must be specified through the option `multiplier`.

`deaths(`*deathsvar*`)` specifies a numeric variable containing the number of observed deaths in each of the age strata.

`pop(`*popvar*`)` designates a numeric variable with the number of persons at risk at each of the age strata (for example, the mid-year population estimate). This option must always be used in conjunction with the previous one whenever option `rates()` is not available nor specified simultaneously.

`by(`*byvar*`)` requests that all computations and outputs, graphics included, are produced for every subgroup defined by the nonmissing categories of one (and only one) numeric variable, *byvar*.

`nyears(`*age_interval_length_var*`)` allows the user to declare a numeric variable specifying the length, in years, of the successive age strata. When this option is not used, it is assumed that all age intervals are equal to one year (in other words, it is assumed by the program that the life table being calculated is a complete life table). The (within subgroup) value for the last interval (an open one) is always (re)set to one.

`radix(#)` allows the user to modify the life table number of persons at risk at exactly age 0; that is, the size of the life table conventional cohort (radix or l_0). The default value, assumed whenever this option is not used, is again 10^5.

`sclist(`*varlist*`)` specifies a list of from one to 20 numeric variables registering the cause-specific number of observed deaths in each of the age strata (and, eventually, for each of the `by()` subgroups). Use of this option will always be followed by a "multiple causes-of-death" analysis, consisting in the output of a number of additional tables. These will include estimates for the 1) at birth, and lifetime (beyond age x) conditional probabilities of death for each specific cause ($\Pr(\text{death by cause } i | \text{age } x)$), 2) absolute risk of dying by each cause, given a certain age, during the following interval (q_{xi}), 3) cumulative distributions of deaths by cause and age (F_{xi}), and 4) probabilities of death after a certain age x, given that the death is caused by each of the specific categories considered. Whenever this option is utilized, it is safer and wiser to also use option `deaths(`*deathsvar*`)` to explicitly declare the total number of deaths observed to occur in each age strata. However, this requirement is not absolute because, in the absence of an explicit declaration of the total number of deaths in each age level (by all causes), the program automatically assumes that the (row) sum of the variables included in option `sclist(`*varlist*`)` equals the just-mentioned total. Of course, if that is not the case, the results may be compromised. So, when the `deaths(`*deathsvar*`)` option is not being used, care must be taken to assure that this assumption holds, for instance by including in the `sclist(`*varlist*`)` option, one variable for a residual specific cause-of-death category corresponding to the remaining, not otherwise explicitly considered, causes of death. Finally, as an alternative to the specification of the absolute number of observed deaths by specific causes, the variables for specific causes of death may also contain previously calculated age (and/or group) specific rates of death. It is only necessary that these rates are all referred to the same number of persons (power of ten) as the ones eventually included in the option `rates(`*ratesvar*`)`, and that another option, `allrx` (signifying all specific causes are in rates) is also used to inform the program of this particular circumstance.

`allrx` declares that the variables included in option `sclist(`*varlist*`)` contain specific rates instead of absolute numbers of deaths by each of the specific causes, which is the default for this program. If this option is used together with option `rates(`*ratesvar*`)`, the user is advised to also specify options `pop(`*popvar*`)`, and/or `deaths(`*deathsvar*`)`, provided the data are available in order to allow the deduction of all the parameters necessary to produce the standard output. If neither of these is available (that is, if strictly only rates are declared), the program will not be able to calculate the potential years of life lost indicators.

`weights(`$\#_1\left[\#_2\left[\#_3\left[\#_4\right]\right]\right]$`)` allows specification of the contribution made for the total time lived on each of the first four life table periods by the cases registered as deaths occurring in these same time intervals. The quantities specified must be expressed in terms of the proportion of the total interval presumably lived by any person ultimately deceased anytime during the same interval. The `weights` option allows the modification of the default values for one, or all, of the first four periods of living. By default, the weights used by `lifetabl` are those used by Selvin (1991) which, apparently, were taken from Chiang (1984). In fact, they are equivalent to an explicit specification of this option as `weights(0.113 0.430 0.450 0.470)`. The variable number of parameters passed to the program is interpreted according to their order, as pertaining respectively to the first, second, third, and fourth age intervals. If fewer than four of these parameters are specified, the program assumes that the omitted parameters are the ones at the end of the list, and attributes to them the default value

of 0.50 (for instance, `w(0.30)` will be interpreted as `weights(0.30 0.50 0.50 0.50)`). Concerning all the time periods following the fourth, it is also always assumed that each death contributes to the total time lived in each age level by the cohort of people alive at its beginning with a time equal to one half of the respective interval length.

`pyll(#)` provides the possibility of modification of the upper age limit utilized in the calculation of the indicator "potential years of life lost" until age #. The default value used by `lifetabl` is `pyll(65)`, following the convention adopted by the U.S. Centers for Disease Control and Prevention, and also by a number of other national and international agencies. In this option, as well as in all outputs of the program, we also complied with the Murray and Lopez (1996) suggestion of always explicitly indicating the upper age limit used in the calculations.

`keep` instructs the program to retain several of the variables generated during the life table construction procedure. By default, `lifetabl` restores the original dataset after processing. However, when option `keep` is included, the user data is left behind, unchanged, and a new datafile, with an added set of variables, but truncated according to any `if` or `in` expressions also utilized in the command, is kept in memory. By default, this new working file is saved under the name `_SaVeD_.dta`, thus replacing any equally named pre-existent file. The variables added to the original set are

`_Rx`	Life table mortality rates, per 10^5 persons
`_qx`	Probability of dying in interval
`_dx`	Expected number of deaths in interval
`_px`	Probability of surviving the interval
`_lx`	Number of people alive at the beginning of the interval
`_Lx`	Cumulative years lived through x
`_Tx`	Total time lived beyond x
`_ExpYL`	Expected years of life at age x
`_Surv`	Survival probability
`_SurvVar`	Greenwood variance for `_Surv`
`_HRate`	Hazard rate

`label(`*labelvar*`)` specifies a string variable to be used in the outputs in the substitution of conventional ordered values in the naming of the successive age levels. In contrast with the truly optional use of this `label` option, the user is always obliged to (also) specify one variable (which may be the same) through option `strata(`*age_level_var*`)`, in order to clearly indicate the correct ordering of the age levels. Due to the output space available, the strings stored in *labelvar* will appear truncated in the output if they are longer than 7 characters.

`multiplier(#)` specifies the number of persons for which the mortality rates declared by the *ratesvar* in option `rates` are referred. Allowed values are powers 0 through 6 of ten. The default value for the rates multiplier, assumed whenever this option is not used, is 10^5.

`not` suppresses the display of tables, which may be useful if the user is mainly interested in the production of graphs or in comparing subgroup life expectancies.

`noo` completely suppresses the written outputs. Following this option, only graphs will be displayed provided they have been requested in the same command line.

`allsct` is used in combination with `sclist(`*varlist*`)` to instruct the program to display three additional tables related with the multiple cause-of-death analysis. These supplementary tables exhibit the observed number of deaths by each cause and age level (D_{xi}), the life table expected number of deaths by each cause at each age level x (d_{xi}), and the life table expected total number of deaths by each cause occurring after each age x (W_{xi}).

`sconly` instructs `lifetabl` to restrict the output only to results related with the multiple causes-of-death analysis.

`noyll` allows the user to request that all output related with the potential years of life lost indicators not be displayed.

`saving(`*filename*`)` allows the user to specify the name of a new filename to save the results. The name for the file must be fewer than eight characters, according to the general naming rules. This option may be combined with `replace` to allow the program to substitute any existing file with the same name.

`replace` instructs `lifetabl` to overwrite any existing file similarly named when `saving(`*filename*`)` is specified.

`grphs` specifies the production of all the five available groups of graphics. The same graphs may instead be produced individually by using the next five options.

`ge` requests the display of a graphed expectation of life function (expectation for the mean years of remaining life to be lived, at each age), for one or more of the subgroups considered.

`gp` requests the graph of the cumulative (proportional) distribution(s) of expected deaths.

`gs` requests the production of a graph with the survival function, or functions, if subgroups are being considered.

`gh` requests a graph of the hazard rate function, or functions, if subgroups are being considered.

`gsc` requests a set of graphs related with the specific causes of death included in option `sclist(`*varlist*`)`.

`lograte` requests a graph of the log hazard rate(s) when the `gh` or `grphs` options are specified.

Examples

The examples provided below use three different data files allowing a complete exploration of the `lifetabl` command. The first dataset, named `lifetabl.dta`, was prepared with data taken from Selvin (1991, 245–248) and contains several variables related to California male and female mortality in 1980. The second file, `mcauses.dta`, stores Portuguese mortality data from several years (1993–1998) disaggregated on sex. This file also includes variables registering the number of observed deaths in each year-sex-age-strata due to some specific causes of death. The third data file included in this insert, `4deaths.dta`, stores data on California male mortality in 1980 related to four specific categories (lung cancer, ischemic heart disease, motor vehicle accidents, and all other causes). It was also prepared with data taken from Selvin (1991, 264) with the purpose of allowing a cross-validation of the `lifetabl` command outputs by replicating the numeric results published by that author.

```
. use lifetabl
(California Mortality: 1980, males & females. Source: Selvin,1991)
. describe
Contains data from lifetabl.dta
  obs:           182                          California Mortality (1980,
                                                males & females. Source:
                                                Selvin,1991)
 vars:            11                          19 Sep 2000 13:35
 size:         7,826 (99.1% of memory free)
-------------------------------------------------------------------------------
   1. strata    byte   %9.0g                  Age strata order
   2. agelab    str7   %9s                    Labels for the age strata
   3. deaths    int    %5.0f                  Number of observed deaths
   4. pop       float  %9.0g                  Resident population
   5. sex       byte   %9.0g       sex        sex
   6. Rx100000  float  %9.0g                  Rate: deaths by 10^5 residents
   7. Rx10000   float  %9.0g                  Rate: deaths by 10^4 residents
   8. Rx1000    float  %9.0g                  Rate: deaths by 10^3 residents
   9. Rx100     float  %9.0g                  Rate: deaths by 10^2 residents
  10. Rx10      float  %9.0g                  Rate: deaths by 10 residents
  11. Rx1       float  %9.0g                  Rate: deaths by 1 resident
-------------------------------------------------------------------------------
Sorted by:  sex  strata
```

The string variable `agelab`, as well as `strata` of byte type, codifies information for the 91 age strata in both sexes.

Suppose we want to produce a complete life table for the male population.

```
. lifetabl if sex==1 , s(strata) r(Rx100000).
---------------------------------- LIFE TABLE ----------------------------------
  Radix: 100000
  Nr of age strata: 91
  Weights (first 4 strata):  .113, .430, .450, .470       Rx multiplier: 100000
-------------------------------------------------------------------------------
  Strata |     Rx        qx        dx        lx        Lx        Ex       Sx
-------------------------------------------------------------------------------
     1   |  1671.3    0.01668     1647     100000      98521     69.61    1.000
     2   |   104.5    0.00104      103      98353      98295     69.77    0.984
     3   |    63.5    0.00063       62      98251      98216     68.84    0.983
     4   |    53.0    0.00053       52      98188      98161     67.89    0.982
     5   |    37.0    0.00037       36      98136      98118     66.92    0.981
     6   |    49.8    0.00050       49      98100      98076     65.95    0.981
     7   |    39.3    0.00039       39      98051      98032     64.98    0.981
     8   |    51.7    0.00052       51      98013      97987     64.00    0.980
     9   |    37.8    0.00038       37      97962      97944     63.04    0.980
    10   |    39.1    0.00039       38      97925      97906     62.06    0.979
 (output omitted)
    80   |  7969.1    0.07664     2640      34443      33123      7.67    0.344
    81   |  8962.7    0.08578     2728      31803      30439      7.27    0.318
    82   | 10319.1    0.09813     2853      29075      27648      6.90    0.291
    83   | 10820.9    0.10265     2692      26222      24876      6.60    0.262
    84   | 11811.1    0.11153     2624      23530      22218      6.30    0.235
    85   | 12856.8    0.12080     2525      20906      19643      6.02    0.209
```

```
  86   | 14290.7    0.13338     2452       18380       17155       5.78     0.184
  87   | 15102.7    0.14042     2237       15929       14810       5.60     0.159
  88   | 14887.5    0.13856     1897       13692       12744       5.43     0.137
  89   | 17599.7    0.16176     1908       11795       10841       5.22     0.118
  90   | 16666.7    0.15385     1521        9887        9126       5.13     0.099
  91   | 20102.6    1.00000     8366        8366       41616       4.97     0.084
-------------------------------------------------------------------------------
  Life Expectancy at birth :  69.607                 (years)
      Crude Mortality Rate :  14.366                 (deaths by 10^3 persons)
```

From left to right the columns exhibit for each of the age strata numbered in the first column: `Rx`, the life table mortality rate; `qx`, the probability of dying in the age period for those alive at its beginning; `dx`, the life table number of expected deaths in each stratum, given the radix; `lx`, the life table number of people alive at the beginning of each age period; `Lx`, the life table total number of years lived in the age period by those alive at its beginning (the life table person-years or stationary population); `Ex`, the expected (mean) years of remaining life for each stratum; and, finally, `Sx`, which informs about the mean probability of surviving until the beginning of each particular stratum.

At the end of the output appears the life table estimate of the crude mortality rate (always expressed in terms of deaths by 10^3 persons), and also the life table estimate of the "at birth" life expectancy statistic.

Notice because we have redundant data in our dataset (observed rates together with number of observed deaths plus population, for each strata), that exactly the same output could have been produced by

```
. lifetabl if sex==1 , s(strata) d(deaths) p(pop)
```

and also by the alternative command lines

```
. lifetabl if sex==1 , s(agelab) d(deaths) p(pop)
. lifetabl if sex==1 , s(agelab) r(Rx100000)
. lifetabl if sex==1 , s(agelab) r(Rx1000) multip(1000)
```

since our string variable `agelab` was constructed in order to reproduce the natural order of the age strata after being sorted. The last example also illustrates the use of observed age-specific mortality rates referred to a number of people other than the 100,000 default value.

Because we have never specified a variable to label the successive age intervals, the age strata were labeled as `1, 2, 3, ... j` (their total). Following the classical division of a population from age 0 to 90+, the strata numbers should be interpreted in this example as $1 = [0 \text{ to } 1[, 2 = [1 \text{ to } 2[, \ldots, 91 = [90 + [$. Notice further that we could also have requested

```
. lifetabl , s(agelab) d(deaths) p(pop) label(agelab) by(sex)
```

thus obtaining 2 life tables, one for each sex, in which the age levels would appear labeled with the contents of the string variable `agelab`.

Until now, it was not necessary to use the option `nyears(`*age_interval_length_var*`)` due to the fact that we have been dealing with complete life tables. Automatically, the program assumes these conditions hold whenever not finding an explicit `nyears(`*age_interval_length_var*`)` option modifying the default assumption.

Let us also consider the request of one or more of the available graphs. It is only necessary to add, to any variation of the commands considered so far, the respective option(s). For example, the command

```
. lifetabl  , s(agelab) r(Rx100000) grphs by(sex)
```

produces, in addition to two life tables for both sex groups (which could have been suppressed provided that we also included in the command line the option `noo`), the following five graphs.

(*Continued on next page*)

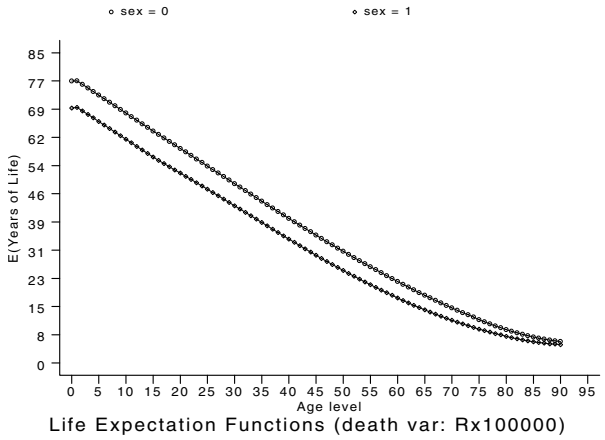

Figure 1. Expectation of life function for `lifetabl` data.

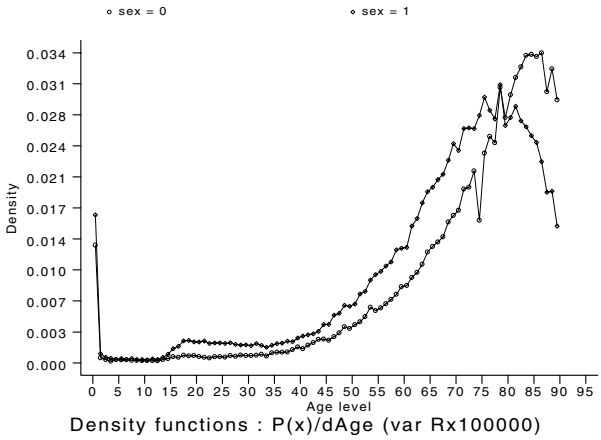

Figure 2. Cause of death density for `lifetabl` data.

Figure 3. Cumulative distribution of expected deaths for `lifetabl` data.

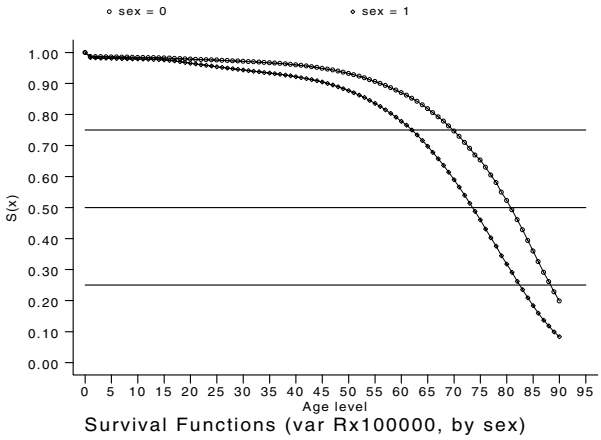

Figure 4. Survival functions for `lifetabl` data.

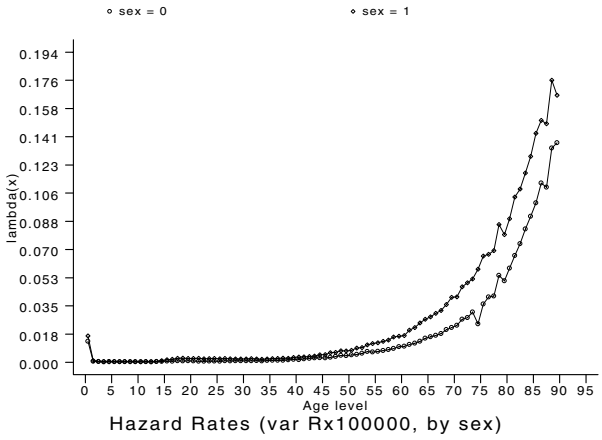

Figure 5.Hazard rate functions for `lifetabl` data.

Example 2

Next, we illustrate the use of `lifetabl` in the context of the analysis of multiple causes of death. We use the `mcauses` dataset.

```
. use mcauses
(Portuguese Mortality in 5 categories: 1993-98, females & males. Source: INE,1999)
```

```
. describe
Contains data from mcauses.dta
  obs:           216                          Portuguese Mortality in 5 categories:
                                                1993-98, females & males.
                                                Source: INE,1999
 vars:            13                          20 Sep 2000 12:34
 size:         6,912 (99.3% of memory free)
-------------------------------------------------------------------------------
   1. year      int    %9.0g                  Calendar year
   2. strata    byte   %9.0g                  Age level order
   3. agelab    str5   %9s                    Age level labels
   4. nyears    byte   %9.0g                  Age level length in years
   5. sex       byte   %9.0g                  Sex : 0 = females; 1 = males
   6. pop       float  %9.0g                  Mid-year population
   7. deaths    int    %9.0g                  Total number of deaths
   8. sc_infec  int    %9.0g                  Deaths by infections
   9. sc_tumor  int    %9.0g                  Deaths by cancer
  10. sc_circ   int    %9.0g                  Deaths by cardiovascular diseases
  11. sc_acid   int    %9.0g                  Deaths by external causes
  12. AllOthrC  int    %9.0g                  deaths-(sum(4 sc vars))
-------------------------------------------------------------------------------
Sorted by:  year  sex  strata
```

Note that some of the variables have exactly the same name and meaning as the similar ones in the data file used previously (`strata`, `agelab`, `sex`, `pop`, `deaths`). However, the data stored in this file is a bit more complex than the data of the `lifetabl.dta` file we used before. Now, we are dealing with data pertaining to several calendar years (1993–1998, coded by the variable `year`), disaggregated by sex, and we also dispose of a set of variables codifying the absolute number of observed deaths related to four diagnostic groups (the `sc_*` variables registering, respectively, infectious diseases, cancer, cardiovascular diseases, and deaths by external causes), plus one other variable (`AllOthrC`) registering deaths linked with aetiologies different from those otherwise explicitly considered. One other feature of this dataset is that the age intervals are bigger than one year in length. In fact, data within each year-sex group is available only for (since birth, successive) five year intervals. Thus, we are not dealing with the standard conditions required for the computation of a complete life table. However, since we do possess one variable codifying the length of each age strata (`nyears`), it is still possible to approximately estimate the life table(s), provided that we use option `nyears(`*varname*`)` to instruct the program that we are dealing with nonequal or nonequal-to-one year age lags.

For instance, if we were (again) simply interested in the calculation of male and female life tables for the calendar year 1995, we would use

```
. lifetabl if year==1995 , s(strata) d(deaths) p(pop) ny(nyears) by(sex) l(agelab)
  Global Life Table
  Results for group: sex = 0
--------------------------------- LIFE TABLE ---------------------------------
  Radix: 100000
  Nr of age strata: 18
  Weights (first 4 strata):  .500, .500, .500, .500       Rx multiplier: 100000
-------------------------------------------------------------------------------
  Strata |     Rx        qx        dx        lx         Lx        Ex       Sx
-------------------------------------------------------------------------------
   00-04 |   165.5    0.00824       165    100000      99588     79.12    1.000
   05-09 |    26.7    0.00133       133     99835      99769     74.27    0.998
   10-14 |    24.1    0.00120       120     99702      99642     69.36    0.997
   15-19 |    37.8    0.00189       188     99582      99488     64.44    0.996
   20-24 |    52.9    0.00264       262     99394      99263     59.56    0.994
   25-29 |    62.9    0.00314       311     99131      98976     54.71    0.991
   30-34 |    78.0    0.00389       385     98820      98628     49.88    0.988
   35-39 |    96.8    0.00483       475     98435      98198     45.06    0.984
   40-44 |   159.8    0.00796       779     97960      97570     40.27    0.980
   45-49 |   209.9    0.01044      1015     97181      96673     35.57    0.972
   50-54 |   322.5    0.01600      1538     96166      95397     30.92    0.962
   55-59 |   493.7    0.02438      2307     94628      93474     26.38    0.946
   60-64 |   769.0    0.03772      3483     92320      90579     21.98    0.923
   65-69 |  1237.9    0.06004      5334     88838      86171     17.74    0.888
   70-74 |  2365.1    0.11165      9323     83504      78843     13.72    0.835
   75-79 |  4494.3    0.20201     14986     74181      66688     10.13    0.742
   80-84 |  8985.2    0.36685     21716     59195      48337      7.06    0.592
     85+ | 21290.5    1.00000     37479     37479     176038      4.70    0.375
-------------------------------------------------------------------------------
  Life Expectancy at birth :  79.125                 (years)
      Crude Mortality Rate :  12.638                 (deaths by 10^3 persons)
```

```
  Global Life Table
  Results for group: sex = 1
-------------------------------- LIFE TABLE ----------------------------------
  Radix: 100000
  Nr of age strata: 18
  Weights (first 4 strata):  .500, .500, .500, .500       Rx multiplier: 100000
------------------------------------------------------------------------------
  Strata |     Rx        qx        dx        lx        Lx        Ex       Sx
------------------------------------------------------------------------------
   00-04 |   207.3   0.01031       206    100000     99484     71.86    1.000
   05-09 |    40.0   0.00200       199     99794     99694     67.03    0.998
   10-14 |    41.5   0.00207       207     99594     99491     62.16    0.996
   15-19 |   123.2   0.00614       610     99388     99083     57.28    0.994
   20-24 |   193.0   0.00960       949     98778     98303     52.62    0.988
   25-29 |   235.7   0.01172      1146     97829     97256     48.10    0.978
   30-34 |   260.5   0.01294      1251     96683     96057     43.64    0.967
   35-39 |   300.4   0.01491      1423     95432     94721     39.18    0.954
   40-44 |   363.7   0.01802      1694     94009     93162     34.74    0.940
   45-49 |   500.7   0.02472      2282     92315     91174     30.33    0.923
   50-54 |   681.2   0.03349      3015     90033     88525     26.04    0.900
   55-59 |  1119.9   0.05447      4740     87017     84647     21.85    0.870
   60-64 |  1768.5   0.08468      6967     82278     78794     17.97    0.823
   65-69 |  2776.5   0.12981      9776     75310     70422     14.40    0.753
   70-74 |  4398.7   0.19815     12985     65534     59041     11.17    0.655
   75-79 |  7144.3   0.30308     15927     52549     44585      8.31    0.525
   80-84 | 12882.5   0.48721     17843     36622     27701      5.84    0.366
     85+ | 24875.8   1.00000     18779     18779     75492      4.02    0.188
------------------------------------------------------------------------------
  Life Expectancy at birth :  71.862                (years)
      Crude Mortality Rate :  13.916                (deaths by 10^3 persons)
```

The successive age strata have now been labeled with the contents of variable `agelab`, and, as anticipated, the program made a background reset of the arguments of the `weights` option to a constant value of 0.50. Nevertheless, this default behavior may always be modified by means of an explicit use of the `weight` option. We had to include an `if` restriction to the data being analyzed because the `mcauses.dta` data file stores figures from several years. Had we preferred to obtain life tables for all the calendar years, and for each of the genders one at a time, we would only have had to change accordingly the `if` expression, as well as the content of the `by` option. Of course, we would also have had to be prepared to deal with a very long list of tables.

Next, we illustrate the study of multiple causes of death by means of life-table-derived methods which provide an opportunity to isolate the impact of specific causes on the pattern of observed human mortality. The command below restricts the scope of the analysis to the observed male gender mortality in 1995.

```
. lifetabl if year==1995 & sex==1 , s(strata) d(deaths) p(pop) ny(nyears)
. > l(agelab) sc(sc_infec sc_tumor sc_circ sc_acid AllOthrC)
```

We present the resulting output, stripped of its first part (a life table equal to the last one in the previous example). Of course, if we had chosen to also include option `sconly` (which stands for "specific causes related output only") in the above command line, we would have achieved a result similar the one shown below.

```
 (output omitted )
  Years of Life Lost (YLL) due to specific causes of death
  ---------------------------------------------------------

  C. of death  |     PYLL(65)      PYLL(65)u       PEYLL        SEYLL
---------------+---------------------------------------------------------
     sc_infec  |        32253         35085        48863        60781
     sc_tumor  |        41440         50945       171222       235453
      sc_circ  |        32795         40945       213076       301393
      sc_acid  |        93360        101450       142275       177100
     AllOthrC  |       108088        121165       270368       359029
---------------+---------------------------------------------------------

  Rates of Years of Life Lost
  ---------------------------

  C. of death  |     PYLL(65)      PYLL(65)u       PEYLL        SEYLL
---------------+---------------------------------------------------------
     sc_infec  |       7.7209         8.3989     199.1563     247.7318
     sc_tumor  |       9.9202        12.1955     697.8684     959.6617
      sc_circ  |       7.8507         9.8017     868.4573    1228.4206
      sc_acid  |      22.3491        24.2857     579.8859     721.8260
     AllOthrC  |      25.8748        29.0052    1101.9686    1463.3340
---------------+---------------------------------------------------------
```

```
  * per 10^3 persons under 65 years: PYLL(65) and PYLL(65)u
  * per 10^3 population members: PEYLL and SEYLL

  Lifetime (at birth) prob. of death
  ----------------------------------

  C. of death  |        Prob.
---------------+---------------
     sc_infec  |      0.02237
     sc_tumor  |      0.21290
      sc_circ  |      0.38801
      sc_acid  |      0.06763
     AllOthrC  |      0.31734
---------------+---------------

  Multiple Causes Life Table(s)
  -----------------------------

  Global and cause-specific probabilities of death in interval
------------------------------------------------------------------------------
                     qx         qx1        qx2        qx3        qx4        qx5
------------------------------------------------------------------------------
  Strata |   Total Risk    sc_infec   sc_tumor    sc_circ    sc_acid   AllOthrC
------------------------------------------------------------------------------
   00-04 |      0.01031     0.00077    0.00026    0.00010    0.00105    0.00813
   05-09 |      0.00200     0.00009    0.00023    0.00004    0.00101    0.00063
   10-14 |      0.00207     0.00009    0.00021    0.00003    0.00094    0.00080
   15-19 |      0.00614     0.00018    0.00035    0.00024    0.00390    0.00148
   20-24 |      0.00960     0.00123    0.00038    0.00022    0.00535    0.00242
   25-29 |      0.01172     0.00267    0.00060    0.00041    0.00497    0.00307
   30-34 |      0.01294     0.00272    0.00092    0.00099    0.00460    0.00371
   35-39 |      0.01491     0.00239    0.00195    0.00164    0.00415    0.00477
   40-44 |      0.01802     0.00145    0.00337    0.00299    0.00429    0.00592
   45-49 |      0.02472     0.00139    0.00654    0.00500    0.00419    0.00761
   50-54 |      0.03349     0.00141    0.00983    0.00797    0.00412    0.01016
   55-59 |      0.05447     0.00120    0.01729    0.01406    0.00514    0.01677
   60-64 |      0.08468     0.00150    0.02740    0.02603    0.00531    0.02443
   65-69 |      0.12981     0.00199    0.03995    0.04576    0.00581    0.03631
   70-74 |      0.19815     0.00218    0.05307    0.07991    0.00755    0.05543
   75-79 |      0.30308     0.00223    0.06668    0.13789    0.00797    0.08831
   80-84 |      0.48721     0.00330    0.08272    0.23738    0.01269    0.15113
     85+ |      1.00000     0.00532    0.11875    0.47913    0.01919    0.37763
------------------------------------------------------------------------------

  Cumulative distributions of deaths (total and by cause)
------------------------------------------------------------------------------
                     Fx         Fx1        Fx2        Fx3        Fx4        Fx5
------------------------------------------------------------------------------
     Age   |   Tot Pr Dth    sc_infec   sc_tumor    sc_circ    sc_acid   AllOthrC
------------------------------------------------------------------------------
       0   |      0.00000     0.00000    0.00000    0.00000    0.00000    0.00000
       5   |      0.00206     0.03426    0.00123    0.00027    0.01545    0.02563
      10   |      0.00406     0.03827    0.00232    0.00036    0.03032    0.02761
      15   |      0.00612     0.04232    0.00332    0.00044    0.04415    0.03013
      20   |      0.01222     0.05016    0.00496    0.00105    0.10141    0.03475
      25   |      0.02171     0.10465    0.00671    0.00162    0.17952    0.04229
      30   |      0.03317     0.22126    0.00944    0.00267    0.25142    0.05175
      35   |      0.04568     0.33897    0.01361    0.00513    0.31719    0.06305
      40   |      0.05991     0.44101    0.02236    0.00917    0.37579    0.07740
      45   |      0.07685     0.50177    0.03726    0.01640    0.43544    0.09494
      50   |      0.09967     0.55893    0.06560    0.02830    0.49262    0.11709
      55   |      0.12983     0.61574    0.10715    0.04679    0.54751    0.14591
      60   |      0.17722     0.66261    0.17782    0.07832    0.61371    0.19190
      65   |      0.24690     0.71787    0.28371    0.13353    0.67829    0.25526
      70   |      0.34466     0.78494    0.42504    0.22234    0.74294    0.34142
      75   |      0.47451     0.84889    0.58839    0.35730    0.81611    0.45589
      80   |      0.63378     0.90134    0.75297    0.54406    0.87803    0.60213
      85   |      0.81221     0.95539    0.89526    0.76811    0.94672    0.77653
------------------------------------------------------------------------------

  Lifetime, and beyond age x, cond. prob. of death: Pr(Dth by cause|x)
------------------------------------------------------------------------------
                     Wx         Px1        Px2        Px3        Px4        Px5
------------------------------------------------------------------------------
     Age   |   Dths aft x    sc_infec   sc_tumor    sc_circ    sc_acid   AllOthrC
------------------------------------------------------------------------------
       0   |       100825     0.02219    0.21115    0.38483    0.06708    0.31475
       5   |        99794     0.02165    0.21307    0.38871    0.06672    0.30985
```

10	99594	0.02160	0.21327	0.38945	0.06585	0.30984
15	99388	0.02156	0.21350	0.39023	0.06504	0.30968
20	98778	0.02151	0.21446	0.39240	0.06152	0.31010
25	97829	0.02048	0.21616	0.39598	0.05672	0.31067
30	96683	0.01802	0.21812	0.40025	0.05236	0.31124
35	95432	0.01550	0.22005	0.40450	0.04839	0.31157
40	94009	0.01330	0.22140	0.40895	0.04491	0.31144
45	92315	0.01207	0.22203	0.41342	0.04136	0.31112
50	90033	0.01096	0.22095	0.41877	0.03811	0.31121
55	87017	0.00988	0.21844	0.42503	0.03517	0.31148
60	82278	0.00917	0.21274	0.43465	0.03175	0.31168
65	75310	0.00838	0.20249	0.44642	0.02889	0.31382
70	65534	0.00734	0.18678	0.46043	0.02653	0.31891
75	52549	0.00643	0.16676	0.47455	0.02367	0.32859
80	36622	0.00603	0.14361	0.48307	0.02252	0.34477
85	18779	0.00532	0.11875	0.47913	0.01919	0.37763

Probability of death after age x given death by cause (all & sp causes)

	Sx	Sx1	Sx2	Sx3	Sx4	Sx5
Age	All causes	sc_infec	sc_tumor	sc_circ	sc_acid	AllOthrC
0	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
5	0.99794	0.96574	0.99877	0.99973	0.98455	0.97437
10	0.99594	0.96173	0.99768	0.99964	0.96968	0.97239
15	0.99388	0.95768	0.99668	0.99956	0.95585	0.96987
20	0.98778	0.94984	0.99504	0.99895	0.89859	0.96525
25	0.97829	0.89535	0.99329	0.99838	0.82048	0.95771
30	0.96683	0.77874	0.99056	0.99733	0.74858	0.94825
35	0.95432	0.66103	0.98639	0.99488	0.68281	0.93695
40	0.94009	0.55899	0.97764	0.99083	0.62421	0.92260
45	0.92315	0.49823	0.96274	0.98360	0.56456	0.90506
50	0.90033	0.44107	0.93440	0.97170	0.50738	0.88291
55	0.87017	0.38426	0.89285	0.95321	0.45249	0.85409
60	0.82278	0.33739	0.82218	0.92168	0.38629	0.80810
65	0.75310	0.28213	0.71629	0.86647	0.32171	0.74474
70	0.65534	0.21506	0.57496	0.77766	0.25706	0.65858
75	0.52549	0.15111	0.41161	0.64270	0.18389	0.54411
80	0.36622	0.09866	0.24703	0.45594	0.12197	0.39787
85	0.18779	0.04461	0.10474	0.23189	0.05328	0.22347

The first section of this output comprises two tables for, respectively, the absolute estimates for potential years of life lost in relation with each of the specific causes of death considered (first table), and the very same statistics transformed in rates per 1,000 persons at risk (second table).

Four different statistics are displayed: 1) PYLL(x), for the potential years of life lost until age x; a sum of the time lags separating the moment of each death (considered to correspond, on average, to the midpoint of each age interval), and a conventional, variable, upper age limit x; 2) PYLL$(x)u$, which is basically the same statistic, but noncorrected (u stands for uncorrected) for the estimated time lived in each age interval by those who die in it (this odd version of the statistic always provides higher estimates than PYLL(x) because it follows the noncredible assumption that all deaths occur at the beginning of each age interval, was only included in the program output in order to allow, by comparison, an evaluation of the impact and the possible bias linked to the discrete nature of the age interval lengths being considered); 3) PEYLL, which stands for period expected years of life lost; a popular alternative to PYLL(x), calculated for the whole life span, and where the years of life lost due to a death occurring at each age interval are made equal to the local expectation of life; and 4) SEYLL, which designates the standard expected years of life lost, a completely different type of indicator, standardized in nature, and thus, from the beginning, conceived with the aim of allowing between-population comparisons (its general definition is similar to that of PEYLL but, differently, the estimates for the age specific expectations of life are substituted from those of a reference population; in our program the "Coale and Demeny West Level 26" standard population, adapted from Murray and Lopez (1996), a population segment which exhibits an at-birth expectation of life of 82.5 years, and whose life expectation structure is applied by our program indifferently to both sexes).

In regard to PYLL(x), we must also add that a wide range of potential limits to life have been used, ranging from 60 to 85 (Murray and Lopez 1996). Although some authors, such as Dempsey (1947) argued that the limit to life should be set equal to the life expectancy at birth for any given population, others have sustained, on the basis of equally weak statistical arguments, that the same limit should be set to some other values, for some bigger, for others lower than life expectancy, no definitive

answer to the problem can be provided. So, while our program by default sets this limit to 65, the user remains free to modify it by means of option `pyll`.

The next table, entitled "Lifetime (at birth) prob. of death" shows an extract from the third of the next four tables presented and grouped together under the general heading "Multiple Causes Life Table(s)". The figures represent the life table estimates for the at-birth probability of dying due to each of the specific causes under study. For example, it can be seen that while the expectation for Portuguese male newborn of a future death related to a cardiovascular problem is almost 39%, its chances for a future death by accident (plus suicide or homicide) round to 7%.

The remaining tables describe, respectively, the risks of dying in each age interval by means of any of the specific causes (q_{xi}), of the cumulative distributions of the deaths (again, for all reasons, and for the specific causes of death at each time, F_{xi}), of the lifetime, and beyond age x, conditional probabilities of death (that is, the probabilities of dying due to a certain cause after reaching any age x, P_{xi}), and finally, of the conditional probabilities of dying after any age provided that death is caused by any of the listed reasons (that is, the survival probabilities conditional on the causes of death, S_{xi}).

Tables with other useful information could also have been requested, had we decided to include the further option `allsct`. Since the interpretation of this kind of output is straightforward, we limit ourselves to showing some other possible examples the user may want to try, namely to illustrate the fact that it is also possible to obtain graphed versions of the probability functions. For instance, the following command line, quite similar to the previous one (only the two new options added, `gsc` and `noo`), would make the program suppress all written output and, instead, produce a series of 12 graphs, all related with the specific causes of death.

```
. lifetabl if year==1995 & sex==1 , s(strata) d(deaths) p(pop) ny(nyears)
. > l(agelab) sc(sc_infec sc_tumor sc_circ sc_acid AllOthrC) gsc noo
```

We show below some of these graphs, which we saved, one at a time, and later reused by means of the command `graph using`. The first four show the risk of death through age functions for the four specific groups of causes being studied. One obvious feature is the obvious relative increase in risk of dying by external causes and infections after adolescence, followed by a posterior decrease in the same risks, during the thirties and forties, and again an increase much later in life. Also characteristically, the corresponding patterns for cancer and cardiovascular diseases are quite different from these; risks increase monotonically mainly after the forties, for both types of conditions, but with a somewhat more delayed pattern for cardiovascular affections.

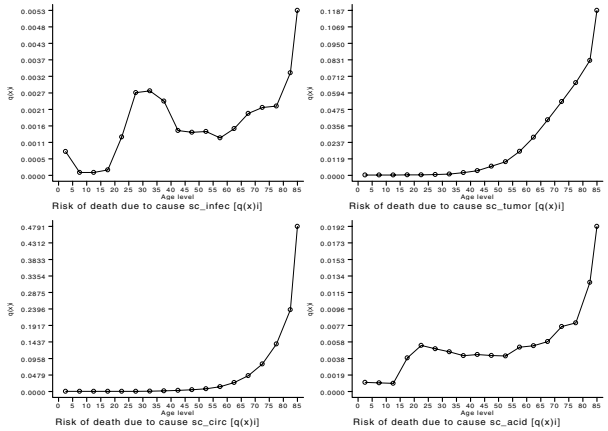

Figure 6. Risk of death through age functions for different groups of causes.

The next two graphs display the cumulative distributions of age at death and the survival probabilities conditional on the cause-of-death for all causes taken together. These two reciprocal sets of functions clearly show (again) that both the external causes of death, and the infectious diseases, detach themselves as important causes of premature death. It is perhaps appropriate to remember that, although these two groups of conditions, even when taken together, sum to a relatively low absolute risk (a global lifetime risk around 9%), they also have already shown to be responsible for a total of 125,613 years of life lost, even when considering a low conventional upper life limit of 65 years (`PYLL(65)`).

(*Continued on next page*)

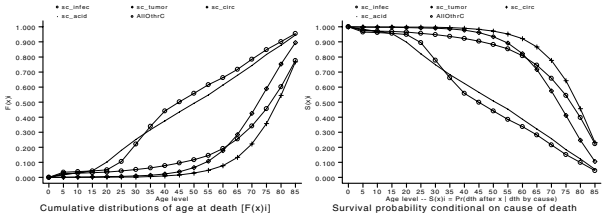

Figure 7. Cumulative distributions of age at death functions.

Finally, we also show a series of four other graphs produced, illustrating the conditional probability of death due to affections related to any of the specific cause groups given age x (that is, the future risk of dying due to any of the causes, for those that already have attained age x).

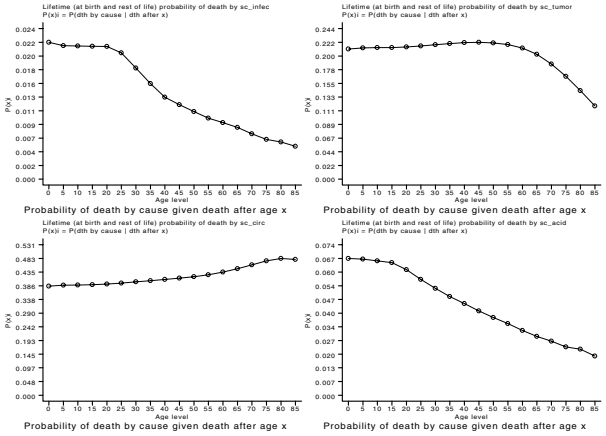

Figure 8. Future risk of dying due to individual causes, given survival to x.

Note that only the relative proportion of dying by means of one of the cardiovascular related affections seems to (definitively) increase monotonically throughout the life span.

The reader may find it useful to explore some further examples we provide below (with outputs omitted) exploring the third database included in this insert, `4deaths.dta`. Once again, most of the examples and illustrations included by Selvin (1991) may be fully reproduced with the `lifetabl` command provided only the user has enough patience and is willing to discount for quite small differences related to computational precision and rounding options.

```
. use 4deaths
. describe
. lifetabl , s(strata) l(agelab) r(Rx100000) ny(nyears)
. lifetabl , s(strata) l(agelab) p(pop) d(deaths) ny(nyears)
. lifetabl , s(strata) l(agelab) p(pop) d(deaths) ny(nyears) noo grphs log
. lifetabl , s(strata) r(Rx100000) ny(nyears) sc(lungcan ihd motor allother) sco
. lifetabl , s(strata) r(Rx100000) ny(nyears) sc(lungcan ihd motor allother) noo gsc
. lifetabl , s(strata) r(Rx100000) ny(nyears) sc(Rlung Rihd Rmotor Rallothr) allrx
. lifetabl , s(strata) r(Rx100000) ny(nyears) sc(Rlung Rihd Rmotor Rallothr) allrx p(pop)
. lifetabl , s(strata) r(Rx100000) ny(nyears) sc(Rlung Rihd Rmotor Rallothr) allrx d(death)
. lifetabl , s(strata) d(deaths) p(pop) ny(nyears) sc(Rlung Rihd Rmotor Rallothr) allrx
. lifetabl , s(strata) d(deaths) p(pop) ny(nyears) sc(Rlung Rihd Rmotor) allrx
. generate Rx1000 = (deaths / pop) * 1000
. label variable Rx1000 "Global death rate, per 1000"
. save 4deaths.dta, replace
. lifetabl , s(agelab) r(Rx1000) ny(nyears) m(1000)
. lifetabl , s(agelab) r(Rx1000) ny(nyears) m(1000) sc(lungcan ihd motor allother) allsct
. lifetabl , s(agelab) r(Rx100000) ny(nyears) sc(lungcan ihd motor allother) grphs noo
. lifetabl , s(agelab) r(Rx100000) ny(nyears)
. lifetabl , s(agelab) r(Rx100000) ny(nyears) radix(10000)
. lifetabl , s(agelab) r(Rx100000) ny(nyears) w(0.3 0.4)
. lifetabl , s(agelab) r(Rx100000) ny(nyears) not gs gh log
```

One final comment on potential years of life lost

Under the (not so realistic) assumption of the possibility of a complete elimination of any chosen cause of death, it is also possible to obtain one other type of summary statistic related to the potential years of life lost concept, but quite different from those described so far. It is only necessary to calculate, one after the other, two life tables: the first, a standard one, and the

second, a slightly modified version in which the total number of deaths variable is reduced by a number equal to the deaths caused by the specific cause chosen. Afterwords, making a simple difference between the two estimates obtained for the expectation of life at birth will generate a statistic which may be interpreted as the mean number of additional years of living gained by each person if the specific cause of death under scrutiny could be eliminated (or, saying it otherwise, a summary measure of the mean number of years of life each one looses due to the existence of that particular cause of death). Consider the example below:

```
. use mcauses
(Portuguese Mortality in 5 categories: 1993-98, females & males. Source: INE,1999)
. lifetabl if sex==1 & year==1995, s(agelab) ny(nyears) p(pop) d(deaths) not
  Global statistics (all causes of death)
------------------------------------------------------------------------------
  Life Expectancy at birth :  71.862                 (years)
      Crude Mortality Rate :  13.916                 (deaths by 10^3 persons)
. gen dthsdiff = deaths - sc_acid
. lifetabl if sex==1 & year==1995, s(agelab) ny(nyears) p(pop) d(dthsdiff) not
  Global statistics (all causes of death)
------------------------------------------------------------------------------
  Life Expectancy at birth :  73.734                 (years)
      Crude Mortality Rate :  13.562                 (deaths by 10^3 persons)
. display 73.734 - 71.862
1.872
```

Thus, according to the 1995 Portuguese data, the complete prevention of all deaths related with accidents, homicides, and suicides (`sc_acid`) would result in an average increase in the life expectation of each male of 1.9 years. It can be verified that similar summary measures for cancer and cardiovascular related deaths amount to, respectively, 2.8 and 5.0 years, while the impact of infectious diseases attains no more than 0.6 years.

Methods and formulas

The all life table is generated from q_x, the calculated risk of dying in each age interval for those alive at its beginning,

$$q_x = \frac{R_x n_x}{1 + n_x w_x R_x}$$

where R_x represents the observed mortality rates, n_x the length of the age interval, and w_x the factor defining the proportion of the interval during which those that die remain, on average, alive. In the last age interval q_x is always made equal to 1.0 because all of those reaching the beginning of that last, open age interval, will die during it. Throughout the program, p_x, the probability of surviving until the end of any age interval for those alive at its beginning is taken from $1 - q_x$.

Survivor function (l_x)

The life table radix, l_0, is set at 100,000. For ages greater than 0, the number of survivors remaining at exactly age x is calculated as

$$l_x = l_{x-1}(1 - q_{x-1}) = l_{x-1} - d_{x-1}$$

Life table decrement function (d_x)

The number of deaths d_x occurring in each age interval (for example, between ages x, and $x + n_x$) is calculated from the survivor function

$$d_x = l_x q_x$$

Notice that d_x in the last interval, d_{xlast}, is equal to l_{xlast} since q_{xlast} is always set to 1.

Stationary population (L_x)

The stationary population is calculated under the assumption that the survivor function declines linearly during each age interval

$$L_x = \frac{(l_x + l_{x+1})}{2} = l_x - d_x + (d_x w_x)$$

Person years lived at and above age x (T_x)

T_x is calculated by summing L_x values at and above age x (of course, at the last age interval, T_x is equal to L_x):

$$T_x = \sum_{i=x}^{\text{xlast}} L_{x_i} L_{x_{i+1}} \cdots l_{x_{\text{xlast}}}$$

Life expectancy at age x (e_x)

Life expectancy at exactly each age x is taken from

$$e_x = \frac{T_x}{l_x}$$

If $x = 0$, then $e_x = e_0$, the at-birth life expectancy.

Life table estimate for the crude mortality rate (R_c)

The life table estimate for this parameter is taken from

$$R_c = \frac{1}{T_0/l_0} = \frac{1}{e_0}$$

Survival probabilities (S_x)

The life table estimate for this parameter is taken from

$$S_x = \frac{l_x}{\text{radix}} = \frac{l_x}{l_0}$$

Hazard rate(s) (HRate)

The life table estimate(s), corresponding to the approximation

$$-\frac{(S_{x+1} - S_x)}{(S_{x+1} + S_x)/2}$$

is (are) estimated by

$$\frac{d_x}{n_x(l_x - w_x d_x)}$$

Potential years of life lost until age x (PYLL(x))

This quantity, which is calculated in reference to a user-defined upper age limit L (that is, $x = L$) is obtained by means of the formula

$$\text{PYLL}(L) = \sum_{x=0}^{L} d_x(L - x_m)$$

where x_m represents the age interval midpoint (that is, $(x + (x + n_x))/2$). Due to the discrete treatment of the age dimension, if the upper age limit L was not defined in order to correspond to any age interval's lower or upper limit, the correct number of deaths for the relevant subsection of the time interval is estimated by assuming a uniform distribution of deaths during the interval.

Period expected years of life lost (PEYLL)

This quantity is computed until the last age to which people survive

$$\text{PEYLL} = \sum_{x=0}^{\text{xlast}} d_x e_x$$

In this formula, e_x represents the age-specific expectation of life.

Standard expected years of life lost (SEYLL)

This summary measure is again computed until the last age to which people survive

$$\text{SEYLL} = \sum_{x=0}^{\text{xlast}} d_x e_x^S$$

where e_x^S represents the standard population age-specific expectation of life. The `lifetabl` command automatically computes this statistic in reference to Coale and Demeny's "West Level 26" age-specific expectations of life for females, which we decided to apply indifferently to every group analyzed. These reference estimates were published by Murray and Lopez (1996), but only for five-year spaced ages from 0 to 100 years. In order to allow their use in the context of complete life tables construction and analysis with our program, we had to estimate values for intermediate years by linear interpolation. The following table presents the final result (the numbers from Murray and Lopez are those for years that are a multiple of five).

Age	e_x	Age	e_x	Age	e_x	Age	e_x	Age	e_x
0	82.50	20	63.08	40	43.53	60	24.83	80	8.90
1	81.84	21	62.10	41	42.57	61	23.95	81	8.36
2	80.87	22	61.12	42	41.61	62	23.07	82	7.83
3	79.90	23	60.13	43	40.64	63	22.20	83	7.29
4	78.92	24	59.15	44	39.68	64	21.32	84	6.76
5	77.95	25	58.17	45	38.72	65	20.44	85	6.22
6	76.96	26	57.19	46	37.77	66	19.59	86	5.83
7	75.97	27	56.21	47	36.83	67	18.74	87	5.43
8	74.97	28	55.23	48	35.88	68	17.90	88	5.04
9	73.98	29	54.25	49	34.94	69	17.05	89	4.64
10	72.99	30	53.27	50	33.99	70	16.20	90	4.25
11	72.00	31	52.29	51	33.07	71	15.42	91	3.98
12	71.00	32	51.31	52	32.14	72	14.63	92	3.71
13	70.01	33	50.34	53	31.22	73	13.85	93	3.43
14	69.01	34	49.36	54	30.29	74	13.06	94	3.16
15	68.02	35	48.38	55	29.37	75	12.28	95	2.89
16	67.03	36	47.41	56	28.46	76	11.60	96	2.71
17	66.04	37	46.44	57	27.55	77	10.93	97	2.53
18	65.06	38	45.47	58	26.65	78	10.25	98	2.36
19	64.07	39	44.50	59	25.74	79	9.58	99	2.18
								100	2.00

Table 1. Coale and Demeny's "West Level 26" age-specific expectations of life for females. Modified from Murray and Lopez (1996, 17).

References

Anderson, R. N. 1999. A Method for constructing complete annual U. S. life tables. National Center for Health Statistics, Vital Health Stat 2(129).

Armitage, P. and G. Berry. 1994. *Statistical Methods in Medical Research*. 3rd ed. Oxford: Blackwell Scientific Publications.

Chiang, C. L. 1984. *The Life Table and its Applications.* Malabar, FL: Krieger.

Gompertz, B. 1825. On the nature of the function expressive of the law of human mortality and on a new mode of determining life contingencies. *Philosophical Transactions of The Royal Society of London* 110: 214–294.

Graunt, J. 1665. *Natural and Political Observations Made Upon the Bills of Mortality*. 3d ed. London: Royal Society.

Halley, E. 1693. An estimate of the degrees of mortality of mankind, drawn from curious tables of the births and funerals at the city of Breslau, with an attempt to ascertain the price of annuities on lives. *Philosophical Transactions of The Royal Society of London* 17: 596–610.

Hill, A. B. and I. D. Hill. 1991. *Bradford Hill's Principles of Medical Statistics.* 12th. ed. London: Arnold.

Murray, C. J. L. and A. D. Lopez. 1996. *The Global Burden of Disease: a Comprehensive Assessment of Mortality and Disability from Diseases, Injuries, and Risk Factors in 1990 and Projected to 2020.* Boston, MA: The Harvard School of Public Health (on behalf of: World Health Organization and World Bank).

Selvin, S. 1991. *Statistical Analysis of Epidemiologic Data.* New York: Oxford University Press.

STB categories and insert codes

Inserts in the STB are presently categorized as follows:

General Categories:

an	announcements	*ip*	instruction on programming
cc	communications & letters	*os*	operating system, hardware, & interprogram communication
dm	data management		
dt	datasets	*qs*	questions and suggestions
gr	graphics	*tt*	teaching
in	instruction	*zz*	not elsewhere classified

Statistical Categories:

sbe	biostatistics & epidemiology	*ssa*	survival analysis
sed	exploratory data analysis	*ssi*	simulation & random numbers
sg	general statistics	*sss*	social science & psychometrics
smv	multivariate analysis	*sts*	time-series, econometrics
snp	nonparametric methods	*svy*	survey sampling
sqc	quality control	*sxd*	experimental design
sqv	analysis of qualitative variables	*szz*	not elsewhere classified
srd	robust methods & statistical diagnostics		

In addition, we have granted one other prefix, *stata*, to the manufacturers of Stata for their exclusive use.

Guidelines for authors

The Stata Technical Bulletin (STB) is a journal that is intended to provide a forum for Stata users of all disciplines and levels of sophistication. The STB contains articles written by StataCorp, Stata users, and others.

Articles include new Stata commands (ado-files), programming tutorials, illustrations of data analysis techniques, discussions on teaching statistics, debates on appropriate statistical techniques, reports on other programs, and interesting datasets, announcements, questions, and suggestions.

A submission to the STB consists of

1. An insert (article) describing the purpose of the submission. The STB is produced using plain TEX so submissions using TEX (or LATEX) are the easiest for the editor to handle, but any word processor is appropriate. If you are not using TEX and your insert contains a significant amount of mathematics, please FAX (979–845–3144) a copy of the insert so we can see the intended appearance of the text.

2. Any ado-files, `.exe` files, or other software that accompanies the submission.

3. A help file for each ado-file included in the submission. See any recent STB diskette for the structure a help file. If you have questions, fill in as much of the information as possible and we will take care of the details.

4. A do-file that replicates the examples in your text. Also include the datasets used in the example. This allows us to verify that the software works as described and allows users to replicate the examples as a way of learning how to use the software.

5. Files containing the graphs to be included in the insert. If you have used STAGE to edit the graphs in your submission, be sure to include the `.gph` files. Do not add titles (e.g., "Figure 1: ...") to your graphs as we will have to strip them off.

The easiest way to submit an insert to the STB is to first create a single "archive file" (either a `.zip` file or a compressed `.tar` file) containing all of the files associated with the submission, and then email it to the editor at `stb@stata.com` either by first using `uuencode` if you are working on a Unix platform or by attaching it to an email message if your mailer allows the sending of attachments. In Unix, for example, to email the current directory and all of its subdirectories:

```
tar -cf - . | compress | uuencode xyzz.tar.Z > whatever
mail stb@stata.com < whatever
```

International Stata Distributors

International Stata users may also order subscriptions to the *Stata Technical Bulletin* from our International Stata Distributors.

Company:	Applied Statistics & Systems Consultants	Countries served:	Israel
		Phone:	+972 (0)6 6100101
Address:	P.O. Box 1169	Fax:	+972 (0)6 6554254
	17100 NAZERATH-ELLIT, Israel	Email:	assc@netvision.net.il
Company:	Axon Technology Company Ltd	Countries served:	Taiwan
Address:	9F, No. 259, Sec. 2	Phone:	+886-(0)2-27045535
	Ho-Ping East Road	Fax:	+886-(0)2-27541785
	TAIPEI 106, Taiwan	Email:	hank@axon.axon.com.tw
Company:	Chips Electronics	Countries served:	Indonesia, Brunei, Malaysia, Singapore
Address:	Kelapa Puyuh IV KB 23	Phone / Fax:	62 - 21 - 452 17 61
	Kelapa Gading Permai	Mobile Phone:	62 - 81 - 884 95 84
	JAKARTA 14240, Indonesia	Email:	puyuh23@indo.net.id
Company:	Dittrich & Partner Consulting	Countries served:	Germany, Austria, Czech Republic, Hungary, Poland
Address:	Kieler Straße 17		
	5. floor	Phone:	+49 2 12 / 26 066 - 0
	D-42697 Solingen, Germany	Fax:	+49 2 12 / 26 066 - 66
URL:	http://www.dpc.de	Email:	sales@dpc.de
Company:	IEM	Countries served:	South Africa, Botswana, Lesotho, Namibia, Mozambique, Swaziland, Zimbabwe
Address:	P.O. Box 2222		
	PRIMROSE 1416, South Africa		
		Phone:	+27-11-8286169
		Fax:	+27-11-8221377
		Email:	iem@hot.co.za
Company:	JasonTech Inc.	Countries served:	Korea
Address:	181-3 Hansang B/D, Bangyidong	Phone:	+82-(0)2-420-6700
	Songpaku	Fax:	+82-(0)2-420-8600
	Seoul 138-050, Korea	Email:	info@jat.co.kr
Company:	MercoStat Consultores	Countries served:	Uruguay, Argentina, Brazil, Paraguay
Address:	9 de junio 1389	Phone:	598-2-613-7905
	CP 11400 MONTEVIDEO, Uruguay	Fax:	598-2-613-7905
		Email:	mercost@adinet.com.uy
Company:	Metrika Consulting	Countries served:	Sweden, Baltic States, Denmark, Finland, Iceland, Norway
Address:	Mosstorpsvagen 48		
	183 30 Taby STOCKHOLM, Sweden	Phone:	+46-708-163128
URL:	http://www.metrika.se	Fax:	+46-8-7924747
		Email:	sales@metrika.se
Company:	MultiON Consulting, SA de CV	Countries served:	Mexico
Address:	Insurgentes Sur 1236-301	Phone:	52 (5) 559-4050 Ext 190
	Mexico, DF, 03200, Mexico	Fax:	52 (5) 559-4048
		Email:	multion@multion.com.mx

(*List continued on next page*)

International Stata Distributors

(*Continued from previous page*)

Company:	Ritme Informatique	Countries served:	France, Belgium, Luxembourg
Address:	34, boulevard Haussmann	Phone:	+33 (0)1 42 46 00 42
	75009 Paris, France	Fax:	+33 (0)1 42 46 00 33
URL:	http://www.ritme.com	Email:	info@ritme.com
Company:	Scientific Solutions S.A.	Countries served:	Switzerland
Address:	Avenue du Général Guisan, 5	Phone:	41 (0)21 711 15 20
	CH-1009 Pully/Lausanne, Switzerland	Fax:	41 (0)21 711 15 21
		Email:	info@scientific-solutions.ch
Company:	Smit Consult	Countries served:	Netherlands
Address:	Doormanstraat 19	Phone:	+31 416-378 125
	5151 GM Drunen, Netherlands	Fax:	+31 416-378 385
URL:	http://www.smitconsult.nl	Email:	info@smitconsult.nl
Company:	Survey Design & Analysis Services Pty Ltd	Countries served:	Australia, New Zealand
Address:	PO Box 1206	Phone:	+61 (0)3 9878 7373
	Blackburn North VIC 3130, Australia	Fax:	+61 (0)3 9878 2345
URL:	http://survey-design.com.au	Email:	sales@survey-design.com.au
Company:	Timberlake Consultants	Countries served:	United Kingdom, Eire
Address:	Unit B3 Broomsleigh Bus. Park	Phone:	+44 (0)208 697 3377
	Worsley Bridge Road	Fax:	+44 (0)208 697 3388
	LONDON SE26 5BN, United Kingdom	Email:	info@timberlake.co.uk
URL:	http://www.timberlake.co.uk		
Company:	Timberlake Consultants Srl	Countries served:	Italy
Address:	Via Baden Powell, 8	Phone:	+39 (0)864 210101
	67039 SULMONA (AQ), Italy	Fax:	+39 (0)864 206014
URL:	http://www.timberlake.it	Email:	timberlake@arc.it
Company:	Timberlake Consulting S.L.	Countries served:	Spain
Address:	Calle Mendez Nunez, 1, 3	Phone:	+34 (9) 5 422 0648
	41011 Sevilla, Spain	Fax:	+34 (9) 5 422 0648
		Email:	timberlake@zoom.es
Company:	Timberlake Consultores, Lda.	Countries served:	Portugal
Address:	Praceta Raúl Brandao, n°1, 1°E	Phone:	351 (0)1 471 73 47
	2720 ALFRAGIDE, Portugal	Fax:	+351 (0)1 471 73 47
		Email:	timberlake.co@mail.telepac.pt
Company:	Vishvas Marketing-Mix Services	Countries served:	India
Address:	C\O S. D. Wamorkar	Phone:	+91-251-440087
	"Prashant" Vishnu Nagar, Naupada	Fax:	+91-22-5378552
	THANE - 400602, India	Email:	vishvas@vsnl.com