STATA TECHNICAL BULLETIN

September 2000

STB-57

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 — page

dm70.1.	Extensions to generate, extended: corrections	2
dm80.1.	Update to changing numeric variables to string	2
dm81.	Utility for time series data	2
sbe19.2.	Update of tests for publication bias in meta-analysis	4
sbe38.	Haplotype frequency estimation using an EM algorithm and log-linear modeling	5
sbe39.	Nonparametric trim and fill analysis of publication bias in meta-analysis	8
sbe40.	Modeling mortality data using the Lee–Carter model	15
sg150.	Hardy–Weinberg equilibrium test in case–control studies	17
sg151.	B-splines and splines parameterized by their values at reference points on the x-axis	20
sg152.	Listing and interpreting transformed coefficients from certain regression models	27
snp15.1.	Update to somersd	35
sts15.	Tests for stationarity of a time series	36
sts16.	Tests for long memory in a time series	39
sts17.	Compacting time series data	44

dm70.1	Extensions to generate, extended: corrections

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

Abstract: Four `egen` functions published in Cox (1999) are corrected; three so that they work properly with long variable lists, and one so that `if` restrictions work properly.

Keywords: `egen`, data management, medians.

A previous insert (Cox 1999) described 24 extra functions for `egen`. This insert contains corrections for four of those.

First, `eqany`, `neqany` and `tag` now work properly when supplied with long variable lists. With each program, as previously published, labeling the variable produced with a list of the variables supplied as arguments would fail with long lists.

Second, `rmed` now works correctly when issued with an `if` restriction. Previously, with an `if` restriction either all or none of the observations were used, depending on whether or not the first observation was selected.

Acknowledgment

I am grateful to Benoit Dulong and Michael Blasnik for alerting me to these problems.

References

Cox, N. J. 1999. dm70: Extensions to `generate`, extended. *Stata Technical Bulletin* 50: 9–17. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, pp. 34–45.

dm80.1	Update to changing numeric variables to string

Nicholas J. Cox, University of Durham, UK, n.j.cox@durham.ac.uk
Jeremy B. Wernow, Stata Corporation, jwernow@stata.com

Abstract: The command `tostring` introduced by Cox and Wernow (2000) has been revised to trap any misguided use of a string format in conversion.

Keywords: string variables, numeric variables, data types, data management.

`tostring` (Cox and Wernow 2000) is a command for changing numeric variables to string. Optionally, users may select a format to use in conversion, which is used as an argument to Stata's `string( )` function (see [U] **16.3.5 String functions**). Such a format should be numeric. For example, the option `format(%7.2f)` specifies that numbers should be rounded to 2 decimal places before conversion to string. However, a user inadvertently specifying a string format would find that the resulting variable would always be an empty string. This is at best a misfeature; hence calls to `format( )` with a string format are now trapped with an error message. The help file has also been revised to make this issue clearer.

Acknowledgment

We are grateful to Kit Baum for alerting us to this problem.

References

Cox, N. J. and J. B. Wernow. 2000. dm80: Changing numeric variables to string. *Stata Technical Bulletin* 56: 8–12.

dm81	Utility for time series data

Christopher F. Baum, Boston College, baum@bc.edu
Vince Wiggins, Stata Corporation, vwiggins@stata.com

Abstract: A program entitled `tsmktim` is described which makes the creation of time variables more convenient.

Keywords: time series, calendar, time variables.

Syntax

`tsmktim` *newtimevar* `,` `start(`*date_literal*`)` [`sequence(`*varname*`)`]

The `tsmktim` command provides a convenient way to generate the appropriate `tsset` command if you do not already have a time variable in the data.

Options

`start(`*date_literal*`)` is required and specifies the starting date for the first observation in the dataset. *date_literal* takes forms such as `1964`, `1999m1`, `1960q1`, `12jan1985`, and so on, depending on whether the data is monthly, quarterly, daily, and so on; see [U] **27 Commands for dealing with dates** for more on how to specify dates.

`sequence(`*varname*`)` specifies that *varname* contains an integer variable that specifies the sequence of the observations. This allows gaps to be specified for the time variable. If the values of *varname* are not sequential, the resulting time variable will have gaps.

Note that the start date specified in `start()` is taken to be the date of the value of the `sequence()` variable in the first observation. If that value is missing, then the date from `start()` is associated with a value of 1 for the `sequence()` variable.

Description

`tsmktim` creates a Stata time variable, *newtimevar*, with an appropriate format for yearly, twice yearly, quarterly, monthly, weekly or daily data, and executes `tsset` to use that variable as the time specifier. Note that the data must be ordered by time before issuing `tsmktim`.

If the `sequence()` option is not specified, the data are assumed to have no gaps and are sequential in the periodicity of *date_literal*, that is, one quarter after another, or one month after another, with no gaps.

Examples

Assume the following data:

```
 x
--
44
21
15
77
.
.
.
```

Then

```
. tsmktim mytime, start(1977q2)
```

will produce

```
 x   mytime
--   ------
44   1977q2
21   1977q3
15   1977q4
77   1978q1
.       .
.       .
.       .
```

while

```
. tsmktim mytime, start(29dec1948)
```

will produce

```
 x      mytime
--   ---------
44   29dec1948
21   30dec1948
15   31dec1948
77   01jan1949
.       .
.       .
.       .
```

and

```
. tsmktim mytime, start(29dec1948) sequence(myseq)
```

will produce

```
 x   myseq     mytime
--   -----   ---------
44      10   29dec1948
21      12   31dec1948
15      13   01jan1949
77      23   11jan1949
.        .     .
.        .     .
.        .     .
```

Saved results

`tsmktim` saves the items returned by `tsset` in `r()`.

sbe19.2	Update of tests for publication bias in meta-analysis

Thomas J. Steichen, RJRT, steicht@rjrt.com

Abstract: Enhancements, changes, and a new option for `metabias` are described.

Keywords: publication bias, meta-analysis.

This insert documents enhancements and changes to `metabias` and provides the syntax needed to use a new feature. A full description of the method and of the operation of the original command and options are given in Steichen (1998). A few revisions were documented later in Steichen et al. 1998. This updated program does not change the implementation of the underlying statistical methodology or modify the original operating characteristics of the program; rather, it follows the syntax changes of Stata version 6.0.

New option

`gweight` requests that the graphic symbols representing the data in the plot be sized proportional to the inverse variance.

Description

The primary purpose of this version is to revise `metabias` to meet and to exploit syntax changes in Stata version 6.0. In addition, some minor deficiencies in the previous implementation have been corrected.

First, `metabias` previously failed to allow a stratified Egger analysis to report a result for the remaining usable strata when one or more strata had undefined slopes, instead it reported a missing overall estimate. Such a limitation is not necessary and has been fixed.

Second, option `gweight` has been added to allow the graphic symbols to be sized according to the data point's inverse variance weight in the optional funnel plot.

Finally, saved results are additionally returned in `r()`. They are documented below.

Saved Results

The system `S_#` macros are unchanged. In addition, the saved results are returned in `r()`. Specifically, `metabias` saves:

S_1	r(k)	number of studies
S_2	r(score)	Begg's score
S_3	r(score_sd)	standard deviation of Begg's score
S_4	r(Begg_p)	Begg's p value
S_5	r(Begg_pcc)	Begg's p, continuity corrected
S_6	r(Egger_bc)	Egger's bias coefficient
S_7	r(Egger_p)	Egger's p value
S_8	r(effect)	overall effect (log scale)

References

Steichen, T. J. 1998. sbe19: Tests for publication bias in meta-analysis. *Stata Technical Bulletin* 41: 9–15. Reprinted in *Stata Technical Bulletin Reprints* vol. 7, pp. 125–133.

Steichen, T. J., M. Egger, and J. Sterne. 1998. sbe19.1: Tests for publication bias in meta-analysis. *Stata Technical Bulletin* 44: 3–4. Reprinted in *Stata Technical Bulletin Reprints* vol. 8, pp. 84–85.

sbe38	Haplotype frequency estimation using an EM algorithm and log-linear modeling

Adrian Mander, MRC Biostatistics Unit, Cambridge, UK, adrian.mander@mrc-bsu.cam.ac.uk

Abstract: This function estimates allele/haplotype frequencies under a log-linear model when phase is unknown. Different log-linear models are compared using a likelihood-ratio test allowing tests for linkage disequilibrium and disease association. These tests can be adjusted for possible confounders in a stratified analysis.

Keywords: Haplotypes, alleles, association studies, stratified analysis, phase unknown, log-linear modeling.

Syntax

`hapipf` *varlist* [`using` *exp*] [`if` *exp*] [, `ldim(`*varlist*`)` `display` `ipf(`*str*`)` `start` `known`

`phase(`*varname*`)` `acc(`*#*`)` `ipfacc(`*#*`)` `nolog` `model(`*#*`)` `lrtest(`*#,#*`)`

`convars(`*str*`)` `confile(`*str*`)`]

Description

This function calculates allele/haplotype frequencies using log-linear modeling embedded within an EM algorithm. The EM algorithm handles the phase uncertainty and the log-linear modeling allows testing for linkage disequilibrium and disease association. These tests can be controlled for confounders using a stratified analysis specified by the log-linear model. The log-linear model can also model the relationship between loci and hence can group similar haplotypes.

The log-linear model is fitted using iterative proportional fitting which is implemented in the `ipf` command introduced in Mander (2000). Note that before `hapipf` can execute, the `ipf` command must be installed. This algorithm can handle very large contingency tables and converges to maximum likelihood estimates even when the likelihood is badly behaved.

The *varlist* consists of paired variables representing the alleles at each locus. If phase is known, then the pairs are the genotypes. When phase is unknown the algorithm assumes Hardy–Weinberg Equilibrium, so models are based on chromosomal data and not genotypic data.

Options

`ldim(`*varlist*`)` specifies the variables that determine the dimension of the contingency table. By default the variables contained in the `ipf` option define the dimension.

`display` specifies whether the expected and imputed haplotype frequencies are shown on the screen.

`ipf(`*str*`)` specifies the log-linear model. It requires special syntax of the form `l1*l2+l3`. This model makes the third locus independent of the first two and includes the interaction between the first and second locus.

`start` specifies that the starting posterior weights of the EM algorithm are chosen at random.

`known` specifies that phase is known.

`phase(`*varname*`)` specifies a variable that contains 1's where phase is known and 0's where phase is unknown.

`acc(`*#*`)` specifies the convergence criteria based on the log likelihood.

`ipfacc(`*#*`)` specifies the convergence criteria for the iterative proportional fitting algorithm.

`nolog` specifies whether the log likelihood is displayed at each iteration.

`model(`*#*`)` specifies a label for the log-linear model being fitted. This label is used in the `lrtest` option.

`lrtest(`*#,#*`)` performs a likelihood-ratio test using two models that have been labeled by the `model` option.

`convars(`*str*`)` specifies a list of variables in the constraints file.

`confile(`*str*`)` specifies the name of the constraints file.

Examples

Data are taken from Sham (1998) that consist of two loci (`a` and `b`), case–control status (`D`) and one stratifying variable (`S`). The first few lines of this dataset are shown below.

```
          a1     a2     b1     b2     D     S
  1.       1      2      1      1     0     0
  2.       1      2      2      2     0     0
  3.       2      2      1      2     1     0
  4.       2      2      1      2     0     0
  5.       1      1      1      2     1     0
  6.       1      1      1      2     1     1
  7.       1      2      2      2     1     0
  8.       1      2      1      1     1     0
  9.       1      2      1      1     1     0
```

Each line represents one subject. When D $= 1$, the subject is a case and when D $= 0$, the subject is a control. Each locus contains pairs of alleles, for locus `a` these are `a1` and `a2`. For example, subject 1 has alleles 1 and 2 at locus `a`. If phase is known, then the ordered genotype would be $1/2$.

If phase is known, the association test between one of the loci and the disease status is the chi-squared test of association in a contingency table. When phase is unknown, the contingency table is not observed, so a model of independence and the saturated model are compared using the likelihood-ratio test. Using the notation first introduced by Wilkinson and Rogers (1973), the independence model is `l1+D` where `l1` is the locus and `D` is the case–control variable and the saturated model is `l1*D`. The commands to do this analysis are

```
. hapipf a1 a2, ipf(l1*D) model(0)
. hapipf a1 a2, ipf(l1+D) model(1) lrtest(0,1)
```

The *varlist* specifies that the alleles at locus `a` are used and corresponds to locus 1 in the `ipf` option.

The test for linkage disequilibrium between two loci is very similar to the test of association between locus and disease status. The models to compare are `l1*l2` and `l1+l2`.

```
. hapipf a1 a2 b1 b2, ipf(l1*l2) model(0)
. hapipf a1 a2 b1 b2, ipf(l1+l2) model(1) lrtest(0,1)
```

Here loci `a` and `b` correspond to loci 1 and 2, respectively, in the `ipf` option.

To obtain the expected haplotype frequencies in the `l1*l2` model requires the `display` option.

```
. hapipf a1 a2 b1 b2, ipf(l1*l2) display
Haplotype Frequency Estimation by EM algorithm
----------------------------------------------
 No. loci          = 2
 Log-Likelihood    = -330.3559939995067
 Df                = 0
 No. parameters    = 4
 No. cells         = 4
Imputed Frequencies
    Haplo        freq       eprob
      1.1   20.150157   .06143341
      1.2   116.84984   .35624952
      2.1   171.84984   .52393245
      2.2   19.150157   .05838463
Expected Frequencies
    Haplo        freq       eprob
      1.1   20.150568   .06143466
      1.2   116.84943   .35624828
      2.1   171.84943   .52393118
      2.2   19.150568   .05838588
```

The haplotypes are listed under the variable `Haplo` and loci are separated by a dot. For a saturated model, the imputed and expected frequencies are the same. For models that are not saturated, the expected frequencies obey the log-linear model. The expected frequencies can be saved as a Stata datafile by the `using` option and this datafile can be used for calculating odds ratios using `tabodds`.

As with normal case–control studies, there is a possibility that the relationship between haplotype/locus and disease is confounded by another variable (`S`). A solution is to perform a stratified analysis using the confounder as the stratifying variable and assuming a common odds model. To test whether this variable is an effect modifier compare the model `l1*l2*S*D` to `l1*l2*S+l1*l2*D+S*D`. The second model assumes that the odds ratios are the same between strata.

```
. hapipf a1 a2 b1 b2, ipf(l1*l2*S*D) model(0)
. hapipf a1 a2 b1 b2, ipf(l1*l2*S+l1*l2*D+S*D) model(1) lrtest(0,1)
```

As there are four possible haplotypes, there are three odds ratios per stratum, which gives three degrees of freedom for the effect modification test.

Grouping haplotypes

For two biallelic loci there are four possible haplotypes. If there is some *a priori* reason that the association is due to only one of the haplotypes, then the effect modification test discussed previously will have lower power than one which groups the other three haplotypes as the comparison group. The grouping allows only one odds ratio per stratum and hence a one degree-of-freedom test. When phase is unknown, the grouping must be performed within the EM algorithm using a constrained log-linear model because the haplotypes are not observed. If the phase was known, this grouping is performed before running `hapipf`.

The relationship between the two odds ratios can be specified by using a constrained log-linear model. The constrained log-linear model uses constraint files and are explained in Mander (2000).

The one degree of freedom test of effect modification is the likelihood-ratio test comparing the common odds ratio model and the model where the odds ratios differ. The base model is `L_1*L_2*S+S*D` and the `L_1*L_2*D` margin is fit using the constraint files. The file (`strata1.dta`) below is the constraint file for the common-odds model. Note that only one odds ratio is freely estimated and all the cells in the `L_1*L_2*D` margin are specified.

	l1	l2	D	Ifreq
1.	1	1	0	1
2.	1	1	1	1
3.	1	2	0	1
4.	1	2	1	1
5.	2	1	0	1
6.	2	1	1	1
7.	2	2	0	1
8.	2	2	1	.

The file (`strata2.dta`) below is the constraint file for the effect modification for one specific haplotype 2.2. All the cells in the `L_1*L_2*D*S` margin are specified and two odds ratios are allowed, the base model is exactly the same.

	l1	l2	D	Ifreq	S
1.	1	1	0	1	0
2.	1	1	0	1	1
3.	1	1	1	1	0
4.	1	1	1	1	1
5.	1	2	0	1	0
6.	1	2	0	1	1
7.	1	2	1	1	0
8.	1	2	1	1	1
9.	2	1	0	1	0
10.	2	1	0	1	1
11.	2	1	1	1	0
12.	2	1	1	1	1
13.	2	2	0	1	0
14.	2	2	0	1	1
15.	2	2	1	.	0
16.	2	2	1	.	1

The following commands obtain the likelihood for both models.

```
. hapipf a1 a2 b1 b2, ipf(S*D+l1*l2*S) confile(strata1) convars(l1 l2 D)
. hapipf a1 a2 b1 b2, ipf(S*D+l1*l2*S) confile(strata2) convars(l1 l2 D S)
```

From the output, the likelihood-ratio test statistic is .82741365 on one degree of freedom which is not significant at the 5% level.

Additional information

There are numerous models that the log-linear model can specify, and a detailed description of these can be found in Mander and Clayton (2000).

References

Mander, A. P. 2000. sbe34: Log-linear modeling using iterative proportional fitting. *Stata Technical Bulletin* 55: 10–12.

Mander, A. P. and D. G. Clayton. 2000. Haplotype analysis in population-based association studies using Stata. *In preparation.*

Sham, P. 1998. *Statistics in Human Genetics.* London: Arnold.

Wilkinson, G. N. and C. E. Rogers. 1973. Symbolic description of factorial models for analysis of variance. *Applied Statistics* 22: 392–399.

sbe39	Nonparametric trim and fill analysis of publication bias in meta-analysis

Thomas J. Steichen, RJRT, steicht@rjrt.com

Abstract: This insert describes `metatrim`, a command implementing the Duval and Tweedie nonparametric "trim and fill" method of accounting for publication bias in meta-analysis. Selective publication of studies, which may lead to bias in estimating the overall meta-analytic effect and in the inferences derived, is of concern when performing a meta-analysis. If publication bias appears to exist, then it is desirable to consider what the unbiased dataset might look like and then to reestimate the overall meta-analytic effect after any apparently "missing" studies are included. Duval and Tweedie's "nonparametric 'trim and fill' method" is an approach designed to meet these objectives.

Keywords: meta-analysis, publication bias, nonparametric, data augmentation.

Syntax

`metatrim` {*theta* { *se_theta* | *var_theta* } | *exp(theta) ll ul* [*cl*]} [`if` *exp*] [`in` *range*]

[`,` {`var` | `ci`} `reffect` `print` `estimat(`{`run` | `linear` | `quadratic`}`)` `eform` `graph`

`funnel` `level(`*#*`)` `idvar(`*varname*`)` `save(`*filename* [`, replace`]`)` *graph_options*]

where $\{a \mid b \mid ...\}$ means choose one and only one of $\{a, b, ...\}$.

Description

`metatrim` performs the Duval and Tweedie (2000) nonparametric "trim and fill" method of accounting for publication bias in meta-analysis. The method, a rank-based data-imputation technique, formalizes the use of funnel plots, estimates the number and outcomes of missing studies, and adjusts the meta-analysis to incorporate the imputed missing data. The authors claim that the method is effective and consistent with other adjustment techniques. As an option, `metatrim` provides a funnel plot of the filled data.

The user provides the effect estimate, *theta*, to `metatrim` as a log risk-ratio, log odds-ratio, or other direct measure of effect. Along with *theta*, the user supplies a measure of *theta*'s variability (that is, its standard error, *se_theta*, or its variance, *var_theta*). Alternatively, the user may provide the exponentiated form, *exp*(*theta*), (that is, a risk ratio or odds ratio) and its confidence interval, (*ll*, *ul*).

The funnel plot graphs *theta* versus *se_theta* for the filled data. Imputed observations are indicated by a square around the data symbol. Guide lines to assist in visualizing the center and width of the funnel are plotted at the meta-analytic effect estimate and at pseudo-confidence-interval limits about that effect estimate (that is, at $theta \pm z \times se\_theta$, where z is the standard normal variate for the confidence level specified by option `level()`).

Options

`var` indicates that *var_theta* was supplied on the command line instead of *se_theta*. Option `ci` should not be specified when option `var` is specified.

`ci` indicates that *exp*(*theta*) and its confidence interval, (*ll*, *ul*), were supplied on the command line instead of *theta* and *se_theta*. Option `var` should not be specified when option `ci` is specified.

`reffect` specifies an analysis based on random-effects meta-analytic estimates. The default is to base calculations on fixed-effects meta-analytic estimates.

`print` requests that the weights used in the filled meta-analysis be listed for each study, together with the individual study estimates and confidence intervals. The studies are labeled by name if the `idvar()` option is specified, or by number otherwise.

`estimat(`{`run` | `linear` | `quadratic`}`)` specifies the estimator used to determine the number of points to be trimmed in each iteration. The user is cautioned that the `run` estimator, R_0, is nonrobust to an isolated negative point, and that the `quadratic` estimator, Q_0, may not be defined when the number of points in the data set is small. The `linear` estimator, L_0, is stable in most situations and is the default.

`eform` requests that the results in the final meta-analysis, and in the `print` option, be reported in exponentiated form. This is useful when the data represent odds ratios or relative risks.

`graph` requests that point estimates and confidence intervals be plotted. The estimate and confidence interval in the graph are derived using fixed- or random-effects meta-analysis, as specified by option `reffect`.

`funnel` requests a filled funnel graph be displayed showing the data, the meta-analytic estimate, and pseudo confidence-interval limits about the meta-analytic estimate. The estimate and confidence interval in the graph are derived using fixed or random-effects meta-analysis, as specified by option `reffect`.

`level(#)` specifies the confidence level percent for the pseudo confidence intervals; the default is 95%.

`idvar(`*varname*`)` indicates the character variable used to label the studies.

`save(`*filename*[`, replace`]`)` saves the filled data in a separate Stata data file. The *filename* is assumed to have extension `.dta` (an extension should not be provided by the user). If *filename* does not exist, it is created. If *filename* exists, an error will occur unless `replace` is also specified. Only three variables are saved: a study id variable and two variables containing the filled *theta* and *se_theta* values. The study id variable, named `id` in the saved file, is created by `metatrim`; but when option `idvar()` is specified, it is based on that id variable. The filled *theta* and *se_theta* variables are named `filled` and `sefill` in the saved file.

graph_options are those allowed with `graph, twoway`, except `ylabel()`, `symbol()`, `xlog`, `ytick` and `gap` are not recognized by `graph`. For `funnel`, the default *graph_options* include `connect(lll..)`, `symbol(iiioS)`, and `pen(35522)` for displaying the meta-analytic effect, the pseudo confidence interval limits (two lines), and the data points, respectively.

Specifying input variables

The individual effect estimates (and a measure of their variability) can be provided to `metatrim` in any of three ways:

1. The effect estimate and its corresponding standard error (the default method):

   `. metatrim` *theta se_theta* `...`

2. The effect estimate and its corresponding variance (note that option `var` must be specified):

   `. metatrim` *theta var_theta*`, var ...`

3. The risk (or odds) ratio and its confidence interval (note that option `ci` must be specified):

   `. metatrim` *exp*(*theta*) *ll ul*`, ci ...`

   where *exp*(*theta*) is the risk (or odds) ratio, *ll* is the lower limit and *ul* is the upper limit of the risk ratio's confidence interval.

   When input method 3 is used, *cl* is an optional input variable that contains the confidence level of the confidence interval defined by *ll* and *ul*:

   `. metatrim` *exp*(*theta*) *ll ul cl*`, ci ...`

   If *cl* is not provided, `metatrim` assumes that a 95% confidence level was reported for each study. *cl* allows the user to combine studies with diverse or non-95% confidence levels by specifying the confidence level for each study not reported at the 95% level. Note that option `level()` does not affect the default confidence level assumed for the individual studies. Values of *cl* can be provided with or without a decimal point. For example, 90 and .90 are equivalent and may be mixed (i.e., 90, .95, 80, .90, etc.). Missing values within *cl* are assumed to indicate a 95% confidence level.

Note that data in binary count format can be converted to the effect format used in `metatrim` by use of program `metan` (Bradburn et al. 1998). `metan` automatically creates and adds variables for *theta* and *se_theta* to the raw dataset, naming them `_ES` and `_seES`. These variables can be provided to `metatrim` using the default input method.

Explanation

Meta-analysis is a popular technique for numerically synthesizing information from published studies. One of the many concerns that must be addressed when performing a meta-analysis is whether selective publication of studies could lead to bias in estimating the overall meta-analytic effect and in the inferences derived from the analysis. If publication bias appears to exist, then it is desirable to consider what the unbiased dataset might look like and then to reestimate the overall meta-analytic effect after any apparently "missing" studies are included. Duval and Tweedie's "nonparametric 'trim and fill' method" is designed to meet these objectives and is implemented in this insert.

An early, visual approach used to assess the likelihood of publication bias and to provide a hint of what the unbiased data might look like was the funnel graph (Light and Pillemer 1984). The funnel graph plotted the outcome measure (effect size) of the component studies against the sample size (a measure of variability). The approach assumed that all studies in the analysis were estimating the same effect. Therefore, the effect estimates should be distributed about the unknown true effect level and their spread should be proportional to their variances. This suggested that, when plotted, small studies should be widely spread about the average effect, and the spread should narrow as sample sizes increase, resulting in a symmetric, funnel-shaped graph. If the graph revealed a lack of symmetry about the average effect (especially if small, negative studies appeared to be absent) then publication bias was assumed to exist.

Evaluation of a funnel graph was a very subjective process, with bias—or lack of bias—residing in the eye of the beholder. Begg and Mazumdar (1994) noted this and observed that the presence of publication bias induced skewness in the plot and a correlation between the effect sizes and their variances. They proposed that a formal test of publication bias could be constructed by examining this correlation. More recently, Egger et al. 1997 proposed an alternative, regression-based test for detecting skewness in the funnel plot and, by extension, for detecting publication bias in the data. Their numerical measure of funnel plot asymmetry also constitutes a formal test of publication bias. Stata implementations of both the Begg and Mazumdar procedure and the Egger et al. procedure were provided in `metabias` (Steichen 1998; Steichen et al. 1998).

However, neither of these procedures provided estimates of the number or characteristics of the missing studies, and neither provided an estimate of the underlying (unbiased) effect. There exist a number of methods to estimate the number of missing studies, model the probability of publication, and provide an estimate of the underlying effect size. Duval and Tweedie list some of these and note that all "are complex and highly computer-intensive to run" and, for these reasons, have failed to find acceptance among meta-analysts. They offer their new method as "a simple technique that seems to meet many of the objections to other methods."

The following sections paraphrase some of the mathematical development and discussion in the Duval and Tweedie paper.

Estimators of the number of suppressed studies

Let $(Y_j, v_j^2),\ j = 1, \dots, n$, be the estimated effect sizes and within-study variances from n observed studies in a meta-analysis, where all such studies attempt to estimate a common global "effect size" Δ. Define the random-effects (RE) model used to combine the Y_j as

$$Y_j = \Delta + \beta_j + \varepsilon_j$$

where $\beta_j \sim N(0, \tau^2)$ accounts for heterogeneity between studies, and $\varepsilon_j \sim N(0, \sigma_j^2)$ is the within-study variability of study j. For a fixed-effects (FE) model, assume $\tau^2 = 0$.

Further, in addition to n observed studies, assume that there are k_0 relevant studies that are not observed due to publication bias. Both the value of k_0, that is, the number of unobserved studies, and the effect sizes of these unobserved studies are unknown and must be estimated.

Now, for any collection X_i, $i = 1, \dots, N$ of random variables, each with a median of zero and sign generated according to an independent set of Bernoulli variables taking values -1 and 1, let r_i denote the rank of $|X_i|$ and

$$W_N^+ = \sum_{X_i > 0} r_i$$

be the sum of the ranks associated with positive X_i. Then W_N^+ has a Wilcoxon distribution.

Assume that among these N random variables, k_0 were suppressed, leaving n observed values. Furthermore, assume that *the suppression has taken place in such a way that the k_0 values of the X_i with the most extreme negative ranks have been suppressed*. (Note: Duval and Tweedie call this their key assumption and present it italicized, as done here, for emphasis. Further, they label the model for an overall set of studies defined in this way as a *suppressed Bernoulli model* and state that it might be expected to lead to a truncated funnel plot.)

Rank again the n observed $|X_i|$ as r_i^* running from 1 to n. Let $\gamma^* \geq 0$ denote the length of the rightmost run of ranks associated with positive values of the observed X_i; that is, if h is the index of the most negative of the X_i and r_h^* is its absolute rank, then $\gamma^* = n - r_h^*$. Define the "trimmed" rank test statistic for the observed n values as

$$T_n = \sum_{X_i > 0} r_i^*$$

Note that though the distributions of γ^* and T_n depend on k_0, the dependence is omitted in this notation. Based on these quantities, define three estimators of k_0, the number of suppressed studies:

$$R_0 = \gamma^* - 1,$$

$$L_0 = \frac{4T_n - n(n+1)}{2n - 1}$$

and

$$Q_0 = n - 1/2 - \sqrt{2n^2 - 4T_n + 1/4}$$

Duval and Tweedie provide the mean and variance of each estimator as follows (the reader should refer to the original paper for the derivation):

$$\mathrm{E}[R_0] = k_0, \qquad \mathrm{var}[R_0] = 2k_0 + 2$$

$$\mathrm{E}[L_0] = k_0 - k_0^2/(2n-1), \qquad \mathrm{var}[L_0] = 16\ \mathrm{var}(T_n)/(2n-1)^2$$

where

$$\mathrm{var}(T_n) = (n(n+1)(2n+1) + 10k_0^3 + 27k_0^2 + 17k_0 - 18nk_0^2 - 18nk_0 + 6n^2k_0)/24$$

and

$$\mathrm{E}[Q_0] \approx k_0 + \frac{2\ \mathrm{var}(T_n)}{((n-1/2)^2 - k_0(2n-k_0-1))^{3/2}}, \qquad \mathrm{var}[Q_0] \approx \frac{4\ \mathrm{var}(T_n)}{(n-1/2)^2 - k_0(2n-k_0-1)}$$

The authors also report that for n large and k_0 of a smaller order than n, then asymptotically:

$$\begin{aligned} \mathrm{E}[R_0] &= k_0, & \mathrm{var}[R_0] &= o(n); \\ \mathrm{E}[L_0] &\sim k_0, & \mathrm{var}[L_0] &\sim n/3; \\ \mathrm{E}[Q_0] &\sim k_0 + 1/6, & \mathrm{var}[Q_0] &\sim n/3. \end{aligned}$$

These results suggest that L_0 and Q_0 should have similar behavior, but the authors report that in practice Q_0 is often larger, sometimes excessively so. They also note that L_0 generally has smaller mean square error than Q_0 when $k_0 \geq n/4 - 2$.

Duval and Tweedie remark that the R_0 *run* estimator is rather conservative and nonrobust to the presence of a relatively isolated negative term at the end of the sequence of ranks. They suggest that the estimators based on T_n seem more robust to such a departure from the suppressed Bernoulli hypothesis. They also note that the Q_0 *quadratic* estimator is defined only when $T_n < n^2/2 + 1/16$, and that simulations show this to be violated quite frequently when the number of studies, n, is small and when the number of suppressed studies, k_0, is large relative to n. These concerns leave the L_0 *linear* estimator as the best all around choice.

Because only whole studies can be trimmed, the estimators are rounded in practice to the nearest nonnegative integer, as follows:

$$R_0^+ = \max\{0, R_0\}$$

$$L_0^+ = \left[\max\left\{0, L_0 + \frac{1}{2}\right\}\right]$$

$$Q_0^+ = \left[\max\left\{0, Q_0 + \frac{1}{2}\right\}\right]$$

where $[x]$ is the integer part of x.

The Iterative trim and fill algorithm

Because the global "effect size" Δ is unknown, the number and position of any missing studies is correlated with the true value of Δ. Therefore, Duval and Tweedie developed an iterative algorithm to estimate these values simultaneously. The algorithm can be used with any of the three estimators of k_0 defined in the previous section (the `metatrim` program allows the user to specify which one is to be used through the `estimat()` option). Likewise, either a fixed-effects or random-effects meta-analysis model can be used to estimate $\widehat{\Delta}^{(l)}$ within each iteration (l) of the algorithm (the default model in `metatrim` is fixed effects, but random effects is used when option `reffect` is specified). Note that the `meta` program of Sharp and Sterne (1997, 1998) is called by `metatrim` to carry out the meta-analysis calculations.

The algorithm proceeds as follows:

1. Starting with values Y_i, estimate $\widehat{\Delta}^{(1)}$ using the chosen meta-analysis model. Construct an initial set of centered values

$$Y_i^{(1)} = Y_i - \widehat{\Delta}^{(1)}, \qquad i = 1, \ldots, n$$

and estimate $\widehat{k}_0^{(1)}$ using the chosen estimator for k_0 applied to the set of values $Y_i^{(1)}$.

2. Let l be the current step number. Remove $\widehat{k}_0^{(l-1)}$ values from the right end of the original Y_i and estimate $\widehat{\Delta}^{(l)}$ based on this trimmed set of $n - \widehat{k}_0^{(l-1)}$ values: $\{Y_1, \ldots, Y_{n-\widehat{k}_0^{(l-1)}}\}$. Construct the next set of centered values
$$Y_i^{(l)} = Y_i - \widehat{\Delta}^{(l)}, \qquad i = 1, \ldots, n$$
and estimate $\widehat{k}_0^{(l)}$ using the chosen estimator for k_0 applied to the set of values $Y_i^{(l)}$.

3. Increment l and repeat step 2 until an iteration L where $\widehat{k}_0^{(L)} = \widehat{k}_0^{(L-1)}$. Assign this common value to be the estimated value $\widehat{k}_0$. Note that in this iteration it will also be true that $\widehat{\Delta}^{(L)} = \widehat{\Delta}^{(L-1)}$.

4. Augment (that is, "fill") the dataset Y with the $\widehat{k}_0$ imputed symmetric values
$$Y_j^* = 2\widehat{\Delta}^{(L)} - Y_{n-j+1}, \qquad j = 1, \ldots, \widehat{k}_0$$
and imputed standard errors
$$\sigma_j^* = \sigma_{n-j+1}, \qquad j = 1, \ldots, \widehat{k}_0$$
Estimate the "trimmed and filled" value of Δ using the chosen meta-analysis method applied to the full augmented dataset $\{Y_1, \ldots, Y_n, Y_1^*, \ldots, Y_{\widehat{k}_0}^*\}$.

Conceptually, this algorithm starts with the observed data, iteratively trims (that is, removes) extreme positive studies from the dataset until the remaining studies do not show detectable deviation from symmetry, fills (that is, imputes into the original dataset) studies that are left-side mirrored reflections (about the center of the trimmed data) of the trimmed studies and, finally, repeats the meta-analysis on the filled dataset to get "trimmed and filled" estimates. Each filled study is assigned the same standard error as the trimmed study it reflects in order to maintain symmetry within the filled dataset.

Example

The method is illustrated with an example from the literature that examines the association between Chlamydia trachomatis and oral contraceptive use derived from 29 case–control studies (Cottingham and Hunter 1992). Analysis of these data with the publication bias tests of Begg and Mazumdar ($p = 0.115$) and Egger et al. ($p = 0.016$), as provided in `metabias`, suggests that publication bias may affect the data. To examine the potential impact of publication bias on the interpretation of the data, `metatrim` is invoked as follows:

```
. metatrim logor varlogor, reffect funnel var
```

The random-effects model and display of the optional funnel graph are requested via options `reffect` and `funnel`. Option `var` is required because the data were provided as log-odds ratios and variances. By default, the linear estimator, L_0, is used to estimate k_0, as no other estimator was requested. `metatrim` provides the following output:

```
Note: option "var" specified.
Meta-analysis
       |  Pooled      95% CI         Asymptotic      No. of
Method |     Est   Lower   Upper  z_value  p_value   studies
-------+----------------------------------------------------
Fixed  |   0.655   0.571   0.738   15.359    0.000     29
Random |   0.716   0.595   0.837   11.594    0.000

Test for heterogeneity: Q= 37.034 on 28 degrees of freedom (p= 0.118)
Moment-based estimate of between studies variance =  0.021

Trimming estimator: Linear
Meta-analysis type: Random-effects model

iteration |  estimate    Tn    # to trim     diff
----------+--------------------------------------
    1     |    0.716    285         5         435
    2     |    0.673    305         6          40
    3     |    0.660    313         7          16
    4     |    0.646    320         7          14
    5     |    0.646    320         7           0

Filled
Meta-analysis
       |  Pooled      95% CI         Asymptotic      No. of
Method |     Est   Lower   Upper  z_value  p_value   studies
-------+----------------------------------------------------
Fixed  |   0.624   0.542   0.705   14.969    0.000     36
Random |   0.655   0.531   0.779   10.374    0.000

Test for heterogeneity: Q= 49.412 on 35 degrees of freedom (p= 0.054)
Moment-based estimate of between studies variance =  0.031
```

`metatrim` first calls program `meta` to perform and report a standard meta-analysis of the original data, showing both the fixed- and random-effects results. These initial results are always reported as *theta* estimates, regardless of whether the data were provided in exponentiated form.

`metatrim` next reports the trimming estimator and type of meta-analysis model to be used in the iterative process, then displays results at each iteration. The `estimate` column shows the value of $\widehat{\Delta}^{(l)}$ at each iteration. As expected, its value at iteration 1 is the same as shown for the random-effects method in the meta-analysis panel, and then decreases in successive iterations as values are trimmed from the data. Column `Tn` reports the T_n statistic, column `# to trim` reports the successive estimates $\widehat{k}_0^{(l)}$ and column `diff` reports the sum of the absolute differences in signed ranks between successive iterations. The algorithm stops when `diff` is zero.

`metatrim` finishes with a call to program `meta` to report an analysis of the trimmed and filled data. Observe that there are now 36 studies, composed of the $n = 29$ observed studies plus the additional $\widehat{k}_0 = 7$ imputed studies. Also note that the estimate of $\widehat{\Delta}$ reported as the random effects pooled estimate for the 36 studies is not the same as the value $\widehat{\Delta}^{(5)}$ shown in the fifth (and final) line of the iteration panel. These values usually differ when the random-effects model is used (because the addition of imputed values change the estimate of τ^2) but are identical always when the fixed-effects model is used.

In summary, `metatrim` adds 7 “missing” studies to the dataset, moving the random-effects summary estimate from $\widehat{\Delta} = 0.716, 95\%$ CI: $(0.595, 0.837)$ to $\widehat{\Delta} = 0.655, 95\%$ CI: $(0.531, 0.779)$. The new estimate, though slightly lower, remains statistically significant; correction for publication bias does not change the overall interpretation of the dataset. Addition of “missing” studies results in an increased variance between studies, the estimate rising from 0.021 to 0.031, and increased evidence of heterogeneity in the dataset, $p = 0.118$ in the observed data versus $p = 0.054$ in the filled data. As expected, when the trimmed and filled dataset is analyzed with the publication bias tests of Begg and Mazumdar and Egger et al. (not shown), evidence of publication bias is no longer observed ($p = 0.753$ and $p = 0.690$, respectively).

The funnel plot (Figure 1), requested via the `funnel` option, graphically shows the final filled estimate of Δ (as the horizontal line) and the augmented data (as the points), along with pseudo confidence-interval limits intended to assist in visualizing the funnel. The plot indicates the imputed data by a square around the data symbol. The filled dataset is much more symmetric than the original data and the plot shows no evidence of publication bias.

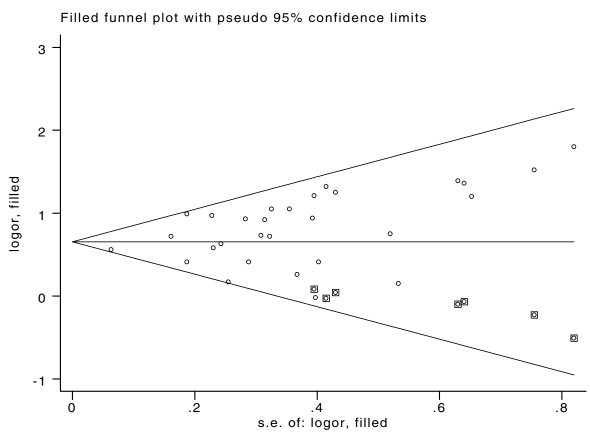

Figure 1. Funnel plot for analysis of Cottingham and Hunter data.

Additional options that can be specified include `print` to show the weights, study estimates and confidence intervals for the filled data set, `eform` to request that the results be reported in exponentiated form in the final meta-analysis and in the `print` option be reported in exponentiated form (this is useful when the data represent odds ratios or relative risks), `graph` to graphically display the study estimates and confidence intervals for the filled data set, and `save(`*filename*`)` to save the filled data in a separate Stata datafile.

Remarks

The Duval and Tweedie method is based on the observation that an unbiased selection of studies that estimate the same thing should be symmetric about the underlying common effect (at least within sampling error). This implies an expectation that the number of studies, and the magnitudes of those studies, should also be roughly equivalent both above and below the common effect value. It is, therefore, reasonable to apply a nonparametric approach to test these assumptions and to adjust the data until the assumptions are met. The price of the nonparametric approach is, of course, lower power (and a concomitant expectation that one may under-adjust the data).

Duval and Tweedie use the symmetry argument in a somewhat roundabout way, choosing to first trim extreme positive studies until the remaining studies meet symmetry requirements. This makes sense when the studies are subject only to publication bias, since trimming should preferably toss out the low-weight, but extreme studies. Nonetheless, if other biases affect the data, in particular if there is a study that is high-weight and extremely positive relative to the remainder of the studies, then the method could fail to function properly. The user must remain alert to such possibilities.

Duval and Tweedie's final step—filling in imputed reflections of the trimmed studies—has no effect on the final trimmed point estimate in a fixed effects analysis but does cause the confidence interval of the estimate to be smaller than that from the trimmed or original data. One could question whether this "increased" confidence is warranted.

The random-effects situation is more complex, as both the trimmed point estimate and confidence interval width are affected by filling, with a tendency for the filled data to yield a point estimate between the values from the original and trimmed data. When the random-effects model is used, the confidence interval of the filled data is typically smaller than that of either the trimmed or original data.

Experimentation suggests that the Duval and Tweedie method trims more studies than may be expected; but because of the increase in precision induced by the imputation of studies during filling, changes in the "significance" of the results occur less often than expected. Thus the two operations (trimming, which reduces the point estimate, and filling, which increases the precision) seem to counter each other.

Another phenomenon noted is a tendency for the heterogeneity of the filled data to be greater than that of the original data. This suggests that the most likely studies to be trimmed and filled are those that are most responsible for heterogeneity. The generality of this phenomenon and its impact on the analysis have not been investigated.

Duval and Tweedie provide a reasonable development based on accepted statistics; nonetheless, the number and the magnitude of the assumptions required by the method are substantial. If the underlying assumptions hold in a given dataset, then, as with many methods, it will tend to under- rather than over-correct. This is an acceptable situation in my view (whereas "over-correction" of publication bias would be a critical flaw).

This author presents the program as an *experimental* tool only. Users must assess for themselves both the amount of correction provided and the reasonableness of that correction. Other tools to assess publication bias issues should be used in tandem. `metatrim` should be treated as merely one of an arsenal of methods needed to fully assess a meta-analysis.

Saved Results

`metatrim` does not save values in the system `S_#` macros, nor does it return results in `r()`.

Note

The command `meta` (Sharp and Sterne 1997, 1998) should be installed before running `metatrim`.

References

Begg, C. B. and M. Mazumdar. 1994. Operating characteristics of a rank correlation test for publication bias. *Biometrics* 50: 1088–1101.

Bradburn, M. J., J. J. Deeks, and D. G. Altman. 1998. sbe24: metan—an alternative meta-analysis command. *Stata Technical Bulletin* 44: 4–15. Reprinted in *The Stata Technical Bulletin Reprints* vol. 8, pp. 86–100.

Cottingham, J. and D. Hunter. 1992. Chlamydia trachomatis and oral contraceptive use: A quantitative review. *Genitourinary Medicine* 68: 209–216.

Duval, S. and R. Tweedie. 2000. A nonparametric "trim and fill" method of accounting for publication bias in meta-analysis. *Journal of the American Statistical Association* 95: 89–98.

Egger, M., G. D. Smith, M. Schneider, and C. Minder. 1997. Bias in meta-analysis detected by a simple, graphical test. *British Medical Journal* 315: 629–634.

Light, R. J. and D. B. Pillemer. 1984. Summing up: The science of reviewing research. Cambridge, MA: Harvard University Press.

Sharp, S. and J. Sterne. 1997. sbe16: Meta-analysis. *Stata Technical Bulletin* 38: 9–14. Reprinted in *The Stata Technical Bulletin Reprints* vol. 7, pp. 100–106.

——. 1998. sbe16.1: New syntax and output for the meta-analysis command. *Stata Technical Bulletin* 42: 6–8. Reprinted in *The Stata Technical Bulletin Reprints* vol. 7, pp. 106–108.

Steichen, T. J. 1998. sbe19: Tests for publication bias in meta-analysis. *Stata Technical Bulletin* 41: 9–15. Reprinted in *The Stata Technical Bulletin Reprints* vol. 7, pp. 125–133.

Steichen, T. J., M. Egger, and J. Sterne. 1998. sbe19.1: Tests for publication bias in meta-analysis. *Stata Technical Bulletin* 44: 3–4. Reprinted in *The Stata Technical Bulletin Reprints* vol. 8, pp. 84–85.

sbe40	Modeling mortality data using the Lee–Carter model

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

Abstract: This article describes the `leecart` command that fits the Lee–Carter model for mortality forecasting. The Lee–Carter model has been a very successful approach for long-term mortality projection and widely applied in demographic studies, as well as actuary. A U.S. mortality dataset is used to illustrate the model estimation.

Keywords: Lee–Carter model, mortality forecasting, singular value decomposition.

The Lee–Carter model

Lee and Carter (1992) proposed a model for forecasting mortality based on the past mortality trends. The model can be written as follows:

$$f_{xt} = \log(m_{xt}) = a_x + b_x k_t + \epsilon_{xt}$$

where m_{xt} is the observed age-specific death rate (ASDR) at age x during time t; a_x, b_x, and k_t are the model's parameters, and ϵ_{xt} is an error term. a_x describes the general age shape of the ASDRs while k_t is an index of the general level of mortality. b_x coefficients describe the tendency of mortality at age x to change when the general level of mortality (k_t) changes.

To estimate the model for a given set of ASDRs (m_{xt}), ordinary least squares can be applied.

The model evidently is underdetermined, which can be seen as follows. Suppose that a, b, k are one solution. Then for any c, $a - bc$, b, $k + c$ also must be a solution. It is also clear that if a, b, k are a solution, then a, bc, k/c also are a solution. Therefore, k is determined only up to a linear transformation, b is determined only up to a multiplicative constant, and a is determined only up to an additive constant. Lee and Carter proposed to normalize the b_x to sum to unity and the k_t to sum to 0, which implies that a_x are simply the averages over time of the $\log(m_{xt})$.

The model cannot be fitted by ordinary regression methods because there are no given regressors; on each side of the equation we have only parameters to be estimated and the unknown index k_t. However, the singular value decomposition (SVD) method can be used to find a least squares solution when applied to the matrix of the logarithms of rates after the averages over time of the (log) age-specific rates have been subtracted (Good 1969). The first right and left vectors and leading value of SVD, after normalization described above, provides a unique solution. The Stata matrix function `matrix svd` is an ideal tool to estimate the parameters in the Lee–Carter model.

Syntax

`leecart` *var_year var_age var_mortality* [`if` *exp*] [`in` *range*]

Description

`leecart` generates three parameter matrices for the Lee–Carter model using the given age-period-specific death rates: a_x, b_x, and k_t. In addition, it yields a matrix of estimated age-specific mortality rates by year.

Examples

U.S. age-period-specific mortality rates from 1900 to 1995 are used below to demonstrate the use of the `leecart` command for the estimation of the Lee–Carter model.

```
. use leecart
(US Death Rates by Year: 1900-1995)
. describe
Contains data from leecart.dta
  obs:        11,520                          US Death Rates by Year:
                                                1900-1995
 vars:             3                          11 Aug 2000 11:11
 size:       184,320 (96.4% of memory free)
-------------------------------------------------------------------------------
   1. year      float  %9.0g                  Year
   2. age       float  %9.0g                  Age
   3. mort      float  %9.0g                  Mortality
-------------------------------------------------------------------------------
Sorted by:
```

```
. summarize
Variable |     Obs        Mean   Std. Dev.       Min        Max
---------+-----------------------------------------------------
    year |   11520      1947.5   27.71251       1900       1995
     age |   11520        59.5   34.64132          0        119
    mort |   11520      .10559   .2098533          0          2

. list in 1/10
          year        age       mort
  1.      1900          0    .145903
  2.      1900          1    .037846
  3.      1900          2     .01886
  4.      1900          3    .013305
  5.      1900          4    .010606
  6.      1900          5    .007799
  7.      1900          6    .005622
  8.      1900          7    .004031
  9.      1900          8    .003057
 10.      1900          9    .002606

. leecart year age m if (age<=5 & year<=1910)

. return list

matrices:
       r(mhat)     : 6 x 11
       r(kt)       : 11 x 1
       r(bx)       : 6 x 1
       r(ax)       : 6 x 1

. mat list r(ax)

r(ax)[6,1]
           ax
a1  -2.0881379
a2  -3.5263281
a3  -4.2174792
a4  -4.5642061
a5  -4.7901406
a6  -5.0804639

. mat list r(bx)

r(bx)[6,1]
           bx
b1  -.52505125
b2    2.043322
b3  -.52963006
b4   .00169509
b5   .00732399
b6   .00234019

. mat list r(kt)

r(kt)[11,1]
            kt
 k1  -.26041863
 k2  -.09171974
 k3  -.06365677
 k4   .00558394
 k5  -.02884234
 k6   .03205825
 k7  -.01836655
 k8   .05870308
 k9   .11228816
k10   .15596412
k11   .09840768

. mat list r(mhat)

r(mhat)[6,11]
             t1          t2          t3          t4          t5          t6
age1  .14207435  .13003127  .12812937   .1235549  .12580853  .12184932
age2  .01727582  .02438612  .02582533  .02975023  .02772939  .03140391
age3  .01691501  .01546924  .01524102  .01469223  .01496257  .01448766
age4  .01042275  .01041977  .01041927  .01041805  .01041866  .01041758
age5  .00832716  .00831687  .00831516  .00831095  .00831304  .00830934
age6  .00621324  .00621569   .0062161  .00621711   .0062166  .00621749

             t7          t8          t9         t10         t11
age1  .12511844  .12015653  .11682305  .11417453  .11767756
age2  .02832935  .03316107  .03699816  .04045184  .03596354
```

```
age3  .01487978  .01428464  .01388494  .01356744  .01398739
age4  .01041847  .01041711  .01041616  .01041539  .01041641
age5  .00831241  .00830772  .00830446   .0083018   .0083053
age6  .00621676  .00621788  .00621866  .00621929  .00621846
. clear
. use leecart
(US Death Rates by Year: 1900-1995)
. leecart year age m if (age<=100 & year<=1995)
. return list
matrices:
          r(mhat)     : 101 x 96
          r(kt)       : 96 x 1
          r(bx)       : 101 x 1
          r(ax)       : 101 x 1
```

Saved Results

`leecart` saves the following matrices in `r()`:

`r(ax)`	a_x
`r(bx)`	b_x
`r(kt)`	k_t
`r(mhat)`	estimated age-specific death rates by year

Acknowledgment

The U.S. mortality dataset was obtained from the Berkeley Mortality Database (http://demog.berkeley.edu/wilmoth/mortality) at the University of California at Berkeley.

References

Lee, R. D. and L. Carter. 1992. Modeling and forecasting the time series of U.S. mortality. *Journal of the American Statistical Association* 87: 659–671.

Good, I. J. 1969. Some applications of the singular value decomposition of a matrix. *Technometrics* 11: 823–831.

sg150	Hardy–Weinberg equilibrium test in case–control studies

Jisheng Cui, University of Melbourne, Australia, j.cui@gpph.unimelb.edu.au

Introduction

Cleves (1999) proposed the Stata command **genhwi** for testing the Hardy–Weinberg equilibrium (HWE) of one sample of individuals. However, in case–control studies, two samples of individuals are collected; the cases and the controls. Usually the genotypic counts of controls are under HWE. We modified **genhwi** to enable it to be applicable to data in case–control studies. The new command **genhwcci** is given for testing whether the genotypic counts of the cases are under HWE, given the controls are under HWE. This test is more powerful than that given by **genhwi** because data from controls are utilized.

Syntax

genhwcci $\#_{\mathrm{AA1}}$ $\#_{\mathrm{Aa1}}$ $\#_{\mathrm{aa1}}$ $\#_{\mathrm{AA2}}$ $\#_{\mathrm{Aa2}}$ $\#_{\mathrm{aa2}}$ [, **label(***genotypes***)** **binvar**]

Description

genhwcci is an immediate command used for estimating allele frequency, genotype frequencies, disequilibrium coefficients, and the associated standard error for codominant traits or data of completely known genotypes in case–control studies. For both genotypic counts of cases and controls, it performs asymptotic HWE tests. It also tests the HWE for genotypic counts of cases, under the assumption that the genotypic counts of controls are under HWE; where $\#_{\mathrm{AA1}}$, $\#_{\mathrm{Aa1}}$, and $\#_{\mathrm{aa1}}$ are the counts for the AA, Aa and aa of the cases; while $\#_{\mathrm{AA2}}$, $\#_{\mathrm{Aa2}}$, and $\#_{\mathrm{aa2}}$ are the genotypic counts of the controls. This command works for biallelic loci only.

Options

`label`(*genotypes*) requests that labels are used in the output of the genotype frequency table.

`binvar` requests that the standard errors from a binomial distribution are reported for each allele frequency. These standard errors are calculated under the assumption that the population is under HWE. By default, standard errors that do not require this assumption are reported.

Remarks

`genhwcci` performs asymptotic tests for HWE for genotypic counts of cases given the controls are under HWE. It also estimates the disequilibrium coefficient (D) and its standard error for genotypic counts of cases and controls, separately. See *Methods and formulas* for details.

Example

Helzlsouer et al. 1998 conducted a case–control study for the association of CYP17 polymorphism and breast cancer, in which 109 cases and 113 controls were collected. We test the hypotheses whether the cases and controls are under HWE, separately. Given the controls are under HWE, we test whether the cases are under HWE. An immediate command is used for testing the HWE.

```
. genhwcci 41 47 21 37 58 18, label(AA AB BB)
    Genotype |         Case         Control  |        Total
-------------+-------------------------------+-------------
          AA |           41              37  |           78
          AB |           47              58  |          105
          BB |           21              18  |           39
-------------+-------------------------------+-------------
       total |          109             113  |          222
        Case
      Allele |      Case     Frequency      Std. Err.
-------------+--------------------------------------
           A |       129        0.5917         0.0350
           B |        89        0.4083         0.0350
-------------+--------------------------------------
       total |       218        1.0000
-------------+--------------------------------------
Estimated disequilibrium coefficient (D) =   0.0260
                                      SE =   0.0232
Hardy-Weinberg Equilibrium Test:
         Pearson chi2 (1) =    1.261  Pr= 0.2614
likelihood-ratio chi2 (1) =    1.258  Pr= 0.2620
 Exact significance prob  =              0.3204
     Control
      Allele |  Control     Frequency      Std. Err.
-------------+--------------------------------------
           A |       132        0.5841         0.0318
           B |        94        0.4159         0.0318
-------------+--------------------------------------
       total |       226        1.0000
-------------+--------------------------------------
Estimated disequilibrium coefficient (D) =  -0.0137
                                      SE =   0.0227
Hardy-Weinberg Equilibrium Test:
         Pearson chi2 (1) =    0.360  Pr= 0.5487
likelihood-ratio chi2 (1) =    0.361  Pr= 0.5481
 Exact significance prob  =              0.6983
Test H0: cases under HWE: (given controls under HWE)
likelihood-ratio chi2 (2) =    1.285  Pr= 0.5260
```

The `label()` option is used to label the genotypes in the table. The order of the genotypes is in the same order as given in the syntax.

For this example, there is no significant evidence that the genotypic counts of cases and controls are not under HWE. Even given the controls are under HWE, there is still no significant evidence that the cases are not under HWE.

Methods and formulas

Here we only give the formulas for testing whether the cases are under HWE, given the controls are under HWE, as the methods for testing one sample has been given by Cleves (1999). The standard error of the disequilibrium coefficient (D) was not included in the command `genhwi`, but it is included in the new command `genhwcci`. Details of the formula can be found in Weir (1990, 74).

The observed case–control data is shown in the following table, where n_i and n'_i represent the number of genotypes among cases and controls, respectively, $i = AA, AB, BB$. Let π_i and π'_i represent the probability that a person has genotype i among cases and controls, respectively. We have $\sum_i \pi_i = 1$ and $\sum_i \pi'_i = 1$

Table 1: Observed genotypic counts

Genotype	Case	Control	Total
AA	n_{AA}	n'_{AA}	m_{AA}
AB	n_{AB}	n'_{AB}	m_{AB}
BB	n_{BB}	n'_{BB}	m_{BB}
Total	n	n'	m

Suppose the controls are randomly selected from the population of interest, which is under HWE. Then the distribution of genotypes in the population is given by

$$\pi'_{AA} = p^2, \qquad \pi'_{AB} = 2pq, \qquad \pi'_{BB} = q^2 \tag{1}$$

where p is the allele frequency of A in the population, and $q = 1 - p$.

Under the null hypothesis, that is, the cases are under HWE, the genotype distribution of the cases is also given by (1) as the controls are assumed to be under HWE. Then the log-likelihood function is given by

$$L_0 = (n_{AA} + n'_{AA})\log(p^2) + (n_{AB} + n'_{AB})\log(2pq) + (n_{BB} + n'_{BB})\log(q^2)$$

with one parameter p.

Under the alternative hypothesis, that is, cases are not under HWE, the genotype distribution of the cases are π_{AA}, π_{AB} and π_{BB}, then the log-likelihood function is given by

$$L_1 = n_{AA}\log\pi_{AA} + n_{AB}\log\pi_{AB} + n_{BB}\log\pi_{BB} + n'_{AA}\log(p^2) + n'_{AB}\log(2pq) + n'_{BB}\log(q^2)$$

with the three parameters p, π_{AA} and π_{AB}, where the π_i sum to one.

To test the null hypothesis versus the alternative hypothesis, we use the statistic $-2(L_0 - L_1)$, which approximates the χ^2 distribution with 2 degrees of freedom. The maximum likelihood estimates of p and π_i can be obtained from the score functions of L_0 and L_1, respectively. More specifically, under the null hypothesis, $\widehat{p} = (2m_{AA} + m_{AB})/(2m)$, while under the alternative hypothesis, $\widehat{p} = (2n'_{AA} + n'_{AB})/(2n')$, $\widehat{\pi}_{AA} = n_{AA}/n$, $\widehat{\pi}_{AB} = n_{AB}/n$, and $\widehat{\pi}_{BB} = n_{BB}/n$. Then $-2(\widehat{L}_0 - \widehat{L}_1)$ is evaluated at the above maximum likelihood estimates and compared with the χ^2 distribution.

Acknowledgment

I would like to thank Dr. Douglas Easton of the University of Cambridge for helpful discussions during his visit to Australia in November of 1999.

References

Cleves, M. 1999. sg110: Hardy–Weinberg equilibrium test and allele frequency estimation. *Stata Technical Bulletin* 48: 34–37. Reprinted in *Stata Technical Bulletin Reprints*, vol. 8, pp. 280–284.

Helzlsouer, K. J. et al. 1998. Association between *CYP17* polymorphisms and the development of breast cancer. *Cancer Epidemiology, Biomarkers & Prevention* 7: 945–949.

Weir, B. S. 1990. *Genetic Data Analysis*. Sunderland, Massachusetts: Sinauer Associates.

sg151	B-splines and splines parameterized by their values at reference points on the x-axis

Roger Newson, Guy's, King's and St Thomas' School of Medicine, London, UK, roger.newson@kcl.ac.uk

Abstract: Two programs are presented for generating a basis of splines in an X-variable, to be used by regression programs to fit spline models. The first, `bspline`, generates a basis of Schoenberg B-splines, which avoid the stability problems associated with plus-functions. The second, `frencurv`, generates a basis of reference splines, whose parameters in the regression model are simply values of the spline at reference points on the x-axis. These programs are complementary to existing spline programs in Stata, and do not supersede them.

Keywords: spline, B-spline, interpolation, quadratic, cubic.

Syntax

`bspline` [*newvarlist*] [`if` *exp*] [`in` *range*] , `xvar(`*varname*`)` [`power(`*#*`)` `knots(`*numlist*`)`

`noexknot` `generate(`*varname*`)` `type(`*type*`)` `labfmt(`*%fmt*`)`]

`frencurv` [*newvarlist*] [`if` *exp*] [`in` *range*] , `xvar(`*varname*`)` [`power(`*#*`)` `refpts(`*numlist*`)` `noexref`

`knots(`*numlist*`)` `noexknot` `generate(`*varname*`)` `type(`*type*`)` `labfmt(`*%fmt*`)`]

Description

`bspline` generates a basis of B-splines in an X-variable, based on a list of knots, for use in fitting a regression model containing a spline in the X-variable. `frencurv` ("French curve") generates a basis of reference splines, for use in fitting a regression model, with the property that the fitted parameters will be values of the spline at a list of reference points on the x-axis. Usually, the regression command is called with the `noconst` option.

Options

`xvar(`*varname*`)` specifies the X-variable on which the splines are based.

`power(`*#*`)`, a nonnegative integer, specifies the power (or degree) of the splines, for example, zero for constant, 1 for linear, 2 for quadratic, 3 for cubic, 4 for quartic, or 5 for quintic. The default is zero.

`knots(`*numlist*`)` specifies a list of at least 2 ascending knots, on which the splines are based. If `knots` are absent, then `bspline` will initialize the list to the minimum and maximum of `xvar`, and `frencurv` will create a list of knots equal to the reference points (in the case of odd-degree splines such as a linear, cubic, or quintic) or midpoints between reference points (in the case of even-degree splines such as constant, quadratic, or quartic).

`noexknot` specifies that the original knot list is not to be extended. If `noexknot` is not specified, then the knot list is extended on the left and right by `power` extra knots on each side, spaced by the distance between the first and last 2 original knots, respectively.

`generate(`*varname*`)` specifies a prefix for the names of the generated splines, which (if there is no *newvarlist*) will be named as *varname1*...*varnameN*, where *N* is the number of splines.

`type(`*type*`)` specifies the storage type of the splines generated (`float` or `double`). If `type` is given as anything else (or not given), then it is set to `float`.

`labfmt(`*%fmt*`)` specifies the format to be used in the variable labels for the generated splines. The default is the format of the `xvar`.

`refpts(`*numlist*`)` (`frencurv` only) specifies a list of at least 2 ascending reference points, with the property that, if the splines are used in a regression model, then the fitted values will be values of the spline at those points. If `refpts` is absent, then the list is initialized to two points, equal to the minimum and maximum of `xvar`.

`noexref` (`frencurv` only) specifies that the original reference list is not to be extended. If `noexref` is not specified, then the reference list is extended on the left and right by $\text{int}(power/2)$ extra reference points on each side, spaced by the distance between the first and last 2 original reference points, respectively.

Remarks

The options described above appear complicated but imply simple defaults for most users. Advanced users and programmers are given the power to specify a comprehensive choice of nondefault splines. The splines are either given the names in the *newvarlist* (if present), or (more usually) generated as a list of numbered variables, prefixed by the `generate` option. (The *newvarlist* is intended mainly for programmers and allows them to store the splines in temporary variables with temporary names.)

Methods and formulas

The principles and definitions of B-splines are given in de Boor (1978) and Ziegler (1969). Practical applications in chemistry are described in Wold (1971, 1974). They are used in signal processing and are associated with a wavelet transformation (Unser, et al. 1992).

Splines are a method of defining models regressing a scalar Y-variate with respect to a scalar X-variate. By definition, a kth-degree spline is defined with reference to a set of q knots $s_1 < s_2 < \ldots < s_q$, dividing the x-axis into intervals of the form $[s_i, s_{i+1})$. In each of those intervals, the regression is a kth-degree polynomial in X (usually a different one in each interval), but the polynomials in any two contiguous intervals have the same jth derivatives at the knot separating the two intervals, for j from zero to $k-1$. By convention, the zeroth derivative is the function itself, so a zeroth-degree spline is simply a right-continuous step function, and a first-degree spline is a simple linear interpolation of values between the knots. (By convention, the intervals $[s_i, s_{i+1})$ are closed on the left and open on the right, but this convention only matters for splines of degree zero, which, by convention, are right-continuous rather than left-continuous.)

Splines can be defined using plus-functions. For a power k and a knot s, the kth-power plus-function at s is defined as

$$P_k(x;s) = \begin{cases} (x-s)^k, & x \geq s \\ 0, & x < s \end{cases} \tag{1}$$

The plus-functions are a basis for the space of splines. That is to say, for any kth-degree spline $S(\cdot)$, with knots $s_1 < s_2 < \ldots < s_q$, there exists a q-vector α such that, for any x,

$$S(x) = \sum_{j=1}^{q} \alpha_j P_k(x; s_j) \tag{2}$$

It might seem that, to fit a spline in a covariate X to a Y-variate, all we have to do is to define a design matrix U, such that $U_{ij} = P_k(x_i; s_j)$ and fit β as a vector of regression coefficients. This is not a good idea for two reasons. First, there are problems with stability, as $P_k(x;s)$ will be very large for $k > 1$ and x much greater than s. Second, the β-parameters estimated will not be easy to explain in words to a nonmathematician. The first problem was solved with the introduction of B-splines by Schoenberg in the 1960s, and these are calculated by `bspline`. The second problem is solved using `frencurv`, which calls `bspline` and then transforms the B-splines, so that the regression parameters will simply be values of the spline at reference points.

The B-splines define an alternative basis of the splines with a given set of knots. Ziegler (1969) defines the B-spline for a set of $k+2$ knots $s_1 < s_2 < \ldots < s_{k+2}$ as

$$B(x; s_1, \ldots, s_{k+2}) = (k+1) \sum_{j=1}^{k+2} \left[\prod_{1 \leq h \leq k+2, h \neq j} (s_h - s_j) \right]^{-1} P_k(x; s_j) \tag{3}$$

The B-spline (3) is positive for x in the open interval (s_1, s_{k+2}) and zero for other x. If the s_j are part of an extended set of knots extending forwards to $+\infty$ and backwards to $-\infty$, then the set of B-splines based on sets of $k+2$ consecutive knots forms a basis of the set of all kth-degree splines defined on the full set of knots. Figure 1 shows the constant, linear, quadratic and cubic B-splines originating at zero and corresponding to unit knots.

For the purposes of `bspline` and `frencurv`, I have taken the liberty of redefining B-splines by scaling the $B(x; s_1, \ldots, s_{k+2})$ in (3) by a factor equal to the mean distance between two consecutive knots to arrive at the scale-invariant B-spline

$$A(x; s_1, \ldots, s_{k+2}) = \frac{s_{k+2} - s_1}{k+1} B(x; s_1, \ldots, s_{k+2}) = \begin{cases} \sum_{j=1}^{k+1} \prod_{h=1}^{k+2} \phi_{jh}(x), & \text{if } s_1 \leq x < s_{k+2} \\ 0, & \text{otherwise} \end{cases}$$

where the functions $\phi_{jh}(\cdot)$ are defined by

$$\phi_{jh}(x) = \begin{cases} 1, & \text{if } h = j \\ (s_{k+2} - s_1)/(s_h - s_j), & \text{if } h = j+1 \\ P_1(x; s_j)/(s_h - s_j), & \text{otherwise} \end{cases} \tag{4}$$

The scaled B-spline $A(x; s_1, \ldots, s_{k+2})$ has the advantage that it is dimensionless, being a sum of products of the dimensionless quantities $\phi_{hj}(x)$. That is to say, it is unaffected by the scale of units of the x-axis, and therefore has the same values, whether x is time in millennia or time in nanoseconds. The original Ziegler B-spline $B(x; s_1, \ldots, s_{k+2})$ is expressed in units of x^{-1}. Therefore, if the scaled B-spline $A(x; s_1, \ldots, s_{k+2})$ appears in a design matrix, then its regression coefficient is expressed in units of the Y-variate, whereas if the original B-spline $B(x; s_1, \ldots, s_{k+2})$ appears in a design matrix, then its regression coefficient is expressed in Y-units multiplied by X-units and will be difficult to interpret, even for a mathematician. The B-splines computed by `bspline` are therefore the $A(x; s_1, \ldots, s_{k+2})$, and users who prefer the original Ziegler B-splines must scale them by $(k+1)/(s_{k+2} - s_1)$. (This factor happens to be one for splines with unit-spaced knots, such as those in Figure 1.)

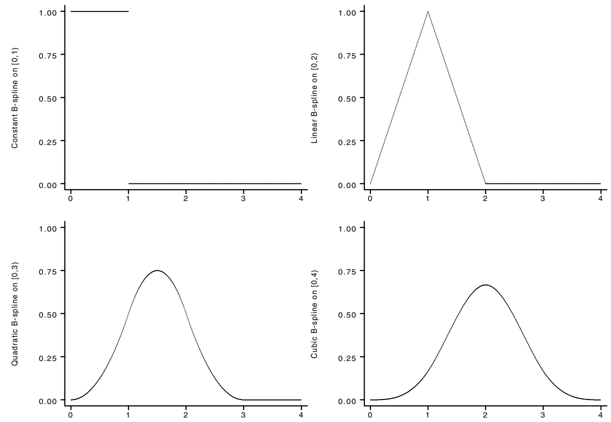

Figure 1. B-splines originating at zero with unit knots.

Given n data points, a Y-variate, an X-covariate, and a set of $q+k+1$ consecutive knots $s_h < \ldots < s_{h+q} < \ldots < s_{h+q+k}$, we can regress the Y-variate with respect to a kth-degree spline in X by defining a design matrix V, with one row for each of the n data points and one column for each of the first q knots, such that

$$V_{ij} = A(x_i; s_{h+j-1}, \ldots, s_{h+j+k}) \tag{5}$$

We can then regress the Y-variate with respect to the design matrix V and compute a vector β of regression coefficients, such that $V\beta$ is the fitted spline. The parameter β_j measures the contribution to the fitted spline of the B-spline originating at the knot s_{h+j-1} and terminating at the knot s_{h+j+k}. There will be no stability problems such as we are likely to have with the original plus-function basis, as each B-spline is bounded and localized in its effect.

It is important to define enough knots. If the sequence of knots $\{s_j\}$ extends to $+\infty$ on the right and to $-\infty$ on the left, then the kth-degree B-splines $A(\cdot; s_{h+j-1}, \ldots, s_{h+j+k})$ on sets of $k+2$ consecutive knots are a basis for the full space of kth-degree splines on the full set of knots. If $S(\cdot)$ is one of these splines, and $[s_j, s_{j+1})$ is an interval between consecutive knots, then the values of $S(x)$ in the interval are affected by the $k+1$ B-splines originating at the knots $s_{j-k}, \ldots, s_j$ and terminating at the knots $s_{j+1}, \ldots, s_{j+k+1}$. It follows that, if we start by specifying a sequence of knots $s_0 < \ldots < s_m$, and we want to fit a spline for values of x in the interval $[s_0, s_m)$, then we must also use k extra knots $s_{-k} < \ldots < s_{-1}$ to the left of s_0 and k extra knots $s_{m+1} < \ldots < s_{m+k}$ to the right of s_m to define the $m+k$ consecutive B-splines affecting $S(x)$ for x in the interval $[s_0, s_m)$. These $m+k$ B-splines originate at the knots $s_{-k}, \ldots, s_{m-1}$ and terminate at the knots $s_1, \ldots, s_{m+k}$, respectively. Any spline $S(\cdot)$, in the full space of kth-degree splines defined using the full set of knots, is equal to a linear combination of these $m+k$ B-splines in the interval $[s_0, s_m)$, which we will denote as the *completeness region* for splines which are linear combinations of these $m+k$ B-splines. These linear combinations are zero for $x < s_{-k}$ and $x \geq s_{m+k}$ and “incomplete” in the outer regions $[s_{-k}, s_0)$ and $[s_m, s_{m+k})$, in which the spline is “returning to zero”.

`bspline` and `frencurv` assume, by default, that the `knots` option specified by the user is only intended to span the completeness region, and that the specified knots correspond to the $s_0, \ldots, s_m$. By default, `bspline` and `frencurv` generate k extra knots on the left, with spacing equal to the difference between the first two knots, and k extra knots on the right, with spacing equal to the difference between the last two knots. If the user specifies the option `noexknot`, then `bspline` assumes that the user has specified the full set of knots, corresponding to $s_{-k}, \ldots, s_{m+k}$ and does not generate any new knots. This allows users to specify their own spacing for the outer knots if they wish but makes the specification of `knots` simpler in the default case because users do not have to count the extra outer knots for themselves.

The B-spline regression parameters are expressed in units of the Y-variable, but they are not easy to interpret. If we have calculated the $n \times q$ matrix V of B-splines as in (5), and we also have a set of q reference X-values $r_1 < r_2 < \ldots < r_q$, then we might prefer to reparameterize the spline by its values at the r_j. To do this, we first calculate a $q \times q$ square matrix W, defined such that

$$W_{ij} = A(r_i; s_{h+j-1}, \ldots, s_{h+j+k}) \tag{6}$$

the value of the jth B-spline at the ith reference point. If β is the (column) q-vector of regression coefficients with respect to the B-splines in V, and γ is the (column) q-vector of values of the spline at the reference points, then

$$\gamma = W\beta \tag{7}$$

If W is invertible, then the n-vector of values of the fitted spline at the data points is

$$V\beta = VW^{-1}W\beta = VW^{-1}\gamma = Z\gamma \tag{8}$$

where $Z = VW^{-1}$ is a transformed $n \times q$ design matrix whose columns contain values of a set of reference splines for the estimation of the reference-point spline values γ.

The choice of reference points is open to the user and constrained mainly by the requirement that the matrix W is invertible. This implies that each of the q B-splines must be positive for at least one of the q reference values, and that each reference value must have at least one positive B-spline value. A natural choice of reference values might be one in the midrange of each B-spline, possibly the central knot for an odd-degree B-spline (such as a linear, cubic, or quintic), or the midpoint between the two central knots for an even-degree B-spline (such as a constant, quadratic, or quartic). This choice has the consequence that, for a spline of degree k, there will be $\texttt{int}(k/2)$ reference points outside the spline's completeness region on the left, and another $\texttt{int}(k/2)$ reference points outside the spline's completeness region on the right. The parameters corresponding to these "extra" reference points will not be easy to explain to nonmathematicians, as they describe the behavior of the spline as it returns to zero outside its completeness region. However, for a quadratic or cubic spline, there is only one such external reference Y-value at each end of the range.

By default (if the user does not specify `knots`), `frencurv` starts with the reference points originally provided (which default to the minimum and maximum of `xvar` if no `refpts` are provided) and chooses knots "appropriately." For an odd-degree spline (`power` odd), the knots are initialized to the original reference points themselves. For an even-degree spline (`power` even), the knots are initialized to midpoints corresponding to the original reference points. That is to say, if there are m original reference points $r_1 < \ldots < r_m$ and `power` is even, then the original knots $s_0 < \ldots < s_m$ are initialized to

$$s_j = \begin{cases} r_1 - (r_2 - r_1)/2, & \text{if } j = 0 \\ (r_j + r_{j+1})/2, & \text{if } 1 \le j \le m-1 \\ r_m + (r_m - r_{m-1})/2, & \text{if } j = m \end{cases} \tag{9}$$

`frencurv` assumes, by default, that the reference points initially provided are all in the completeness region. It adds $\texttt{int}(k/2)$ extra reference points to the left, spaced by the difference between the first two original reference points, and $\texttt{int}(k/2)$ extra reference points to the right, spaced by the difference between the last two original reference points, where k is specified by the `power` option. If `noexref` is specified, then the original `refpts` are assumed to be the complete set, and it is the user's responsibility to choose sensible ones. In either case, the original knots are extended on the left and right as described above, unless `noexknot` is specified. (These rules seem complicated, but lead to sensible defaults if the naive user specifies a list of reference points and expects them to be in the completeness region of the spline, while preserving the ability of advanced users to specify exactly what they want at their own risk.)

Figure 2 shows the constant, linear, quadratic and cubic reference splines corresponding to a reference point at 4, assuming unit reference points and default knots (equal to reference points for odd degree and inter-reference midpoints for even degree). Note that each spline is one at its own reference point and zero at all other reference points. They are similar to (but not the same as) the B-spline wavelets of Unser et al. 1992.

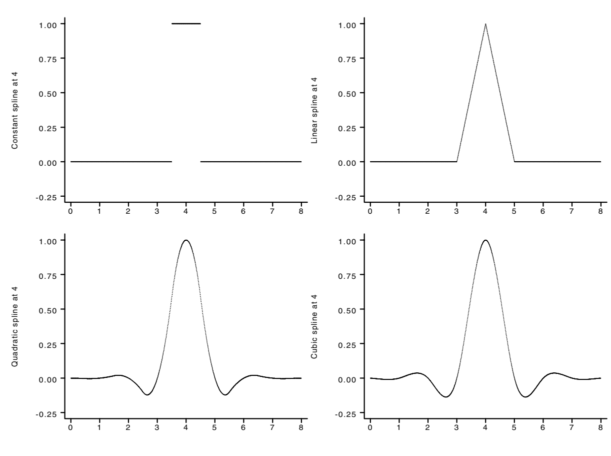

Figure 2. Reference splines at 4 with unit reference points.

Example

In Stata's `auto` data, we can use `frencurv` and `regress` (with the `noconstant` option) to fit a cubic spline for miles per gallon with respect to weight:

```
. frencurv,xvar(weight) refpts(1760(770)4840) gen(cs) power(3)
. describe cs*
  13. cs1       float  %8.4f                Spline at 990 (INCOMPLETE)
  14. cs2       float  %8.4f                Spline at 1,760
  15. cs3       float  %8.4f                Spline at 2,530
  16. cs4       float  %8.4f                Spline at 3,300
  17. cs5       float  %8.4f                Spline at 4,070
  18. cs6       float  %8.4f                Spline at 4,840
  19. cs7       float  %8.4f                Spline at 5,610 (INCOMPLETE)
. regress mpg cs*,noconst robust
Regression with robust standard errors                 Number of obs =      74
                                                       F(  7,    67) =  618.91
                                                       Prob > F      =  0.0000
                                                       R-squared     =  0.9792
                                                       Root MSE      =  3.3469
------------------------------------------------------------------------------
         |               Robust
     mpg |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
     cs1 |   11.82559   15.56642      0.760   0.450      -19.24512    42.89629
     cs2 |   29.21133   1.761704     16.581   0.000       25.69495    32.72771
     cs3 |   22.65796   .7625134     29.715   0.000       21.13597    24.17994
     cs4 |    19.4749    .610094     31.921   0.000       18.25715    20.69266
     cs5 |   15.51593   .8409023     18.452   0.000       13.83748    17.19437
     cs6 |   10.60747   1.585487      6.690   0.000       7.442828    13.77212
     cs7 |  -28.19347   21.59599     -1.305   0.196      -71.29924    14.91229
------------------------------------------------------------------------------
```

We have arbitrarily chosen the reference points to be equally spaced from the minimum of `weight` (1,760 pounds) to the maximum of `weight` (4,840 pounds). By default, `frencurv` ensures that the spline is complete in the range of X-values spanned by the original reference points provided by the user. The `describe` command lists the reference splines with their labels. Note that `frencurv` has added two extra reference points outside the spline's completeness region (at weights of 990 and 5,610 pounds) and indicated this incompleteness in the variable labels. The coefficients fitted by `regress` (with the `noconstant` option) are simply the fitted values of `mpg` at the reference points. Note that the ones corresponding to the splines `cs2` to `cs6` have "sensible" values, corresponding to the expected levels of `mpg` at the appropriate value of `weight`, whereas the ones corresponding to `cs1` and `cs7` have "nonsense" values because they correspond to reference "weights" extrapolated off the edge of the range of sensible `weight` values. This is the price we pay for making all reference points equal to knots of the cubic spline. Figure 3 shows observed and fitted values of `mpg` plotted against `weight`. The fitted curve is calculated using `predict` (see [R] **predict**) and is interpolated cubically between the reference points.

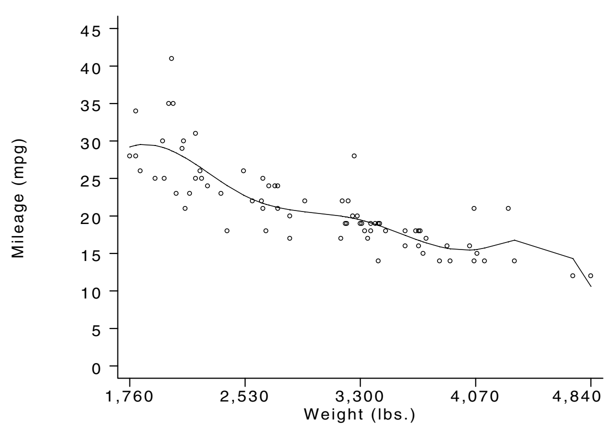

Figure 3. Mileage plotted against weight (points) with fitted cubic spline (line).

The `frencurv` parameterization allows us to use `lincom` to calculate confidence intervals for differences (or other contrasts) between the values of the spline at different reference points. Here, we estimate the difference between expected mileage at weights of 2,530 and 4,070 pounds:

```
. lincom cs3-cs5
 ( 1)  cs3 - cs5 = 0.0
------------------------------------------------------------------------------
     mpg |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
     (1) |   7.142029   1.058829      6.745   0.000       5.028598     9.25546
------------------------------------------------------------------------------
```

We see that cars weighing 2,530 pounds are expected to travel 5.03 to 9.26 more miles per gallon than cars weighing 4,070 pounds.

We can, instead, choose an alternative set of reference points, using **noexref** and specifying our own **knots**. The initial knots are the same initial knots as in the previous model (where they were also reference points), namely 5 equally spaced values from the minimum to the maximum of **weight**. However, the new reference points are $7(=5+2)$ equally spaced values covering the same range. The knots and the reference points are therefore out of synchrony, but the reference points are now all in the completeness region of the spline because they are in the range spanned by the initial knots. (Remember that by default, **bspline** and **frencurv** add new knots on the left and right to make the spline complete over the range of the original knots.) The model is exactly the same model as before (because a spline model is defined by the knots), but the parameters are now *all* sensible within-range **mpg** values, which nontechnical people can understand. Note that we have used **labfmt** to handle the noninteger values of the reference points in the variable labels.

```
. frencurv,xvar(weight) refpts(1760(513.33333)4840) noexr k(1760(770)4840) gen(sp)
> power(3) labfmt(%7.2f)
. describe sp*
  20. sp1       float  %8.4f                  Spline at 1760.00
  21. sp2       float  %8.4f                  Spline at 2273.33
  22. sp3       float  %8.4f                  Spline at 2786.67
  23. sp4       float  %8.4f                  Spline at 3300.00
  24. sp5       float  %8.4f                  Spline at 3813.33
  25. sp6       float  %8.4f                  Spline at 4326.67
  26. sp7       float  %8.4f                  Spline at 4840.00
```

(*Continued on next page*)

```
. regress mpg sp*,noconst robust

Regression with robust standard errors                 Number of obs =      74
                                                       F(  7,    67) =  618.91
                                                       Prob > F      =  0.0000
                                                       R-squared     =  0.9792
                                                       Root MSE      =  3.3469

------------------------------------------------------------------------------
         |               Robust
     mpg |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
     sp1 |   29.21133   1.761704     16.581   0.000       25.69495    32.72771
     sp2 |   25.89924   1.073405     24.128   0.000       23.75671    28.04177
     sp3 |   20.98226   .7479685     28.052   0.000        19.4893    22.47521
     sp4 |    19.4749    .610094     31.921   0.000       18.25715    20.69266
     sp5 |   15.97982   .5560974     28.736   0.000       14.86985     17.0898
     sp6 |   16.74691   1.934879      8.655   0.000       12.88487    20.60894
     sp7 |   10.60747   1.585487      6.690   0.000       7.442828    13.77212
------------------------------------------------------------------------------
```

Finally, for technical people, we can fit the same model yet again, using `bspline` instead of `frencurv`. Here, the splines are B-splines rather than reference splines. The variable labels show the range of positive values of each B-spline, delimited by knots, including the extra knots calculated by `bspline`. The parameters are expressed in miles per gallon but are not easy for nonmathematicians to interpret.

```
. bspline,xvar(weight) knots(1760(770)4840) gen(bs) power(3) labf(%4.0f)

. describe bs*
  27. bs1       float  %8.4f                B-spline on [-550,2530)
  28. bs2       float  %8.4f                B-spline on [220,3300)
  29. bs3       float  %8.4f                B-spline on [990,4070)
  30. bs4       float  %8.4f                B-spline on [1760,4840)
  31. bs5       float  %8.4f                B-spline on [2530,5610)
  32. bs6       float  %8.4f                B-spline on [3300,6380)
  33. bs7       float  %8.4f                B-spline on [4070,7150)

. regress mpg bs*,noconst robust

Regression with robust standard errors                 Number of obs =      74
                                                       F(  7,    67) =  618.91
                                                       Prob > F      =  0.0000
                                                       R-squared     =  0.9792
                                                       Root MSE      =  3.3469

------------------------------------------------------------------------------
         |               Robust
     mpg |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
     bs1 |   8.530818    24.5484      0.348   0.729        -40.468    57.52964
     bs2 |   36.83022   5.330421      6.909   0.000       26.19066    47.46979
     bs3 |   19.41627   2.252816      8.619   0.000       14.91963    23.91291
     bs4 |   21.45246   1.708278     12.558   0.000       18.04272     24.8622
     bs5 |   11.62333   2.241923      5.185   0.000       7.148434    16.09823
     bs6 |   25.14979   7.910832      3.179   0.002       9.359707    40.93988
     bs7 |  -48.57765   34.29427     -1.416   0.161      -117.0293    19.87399
------------------------------------------------------------------------------
```

Technical note

There are other programs in Stata to generate splines. `mkspline` (see [R] **mkspline**) generates a basis of linear splines to be used in a design matrix, as does `frencurv, power(1)`, but the basis is slightly different because the fitted parameters for `frencurv` are reference values, whereas the fitted parameters for `mkspline` are the local slopes of the spline in the inter-knot intervals. `spline` and `spbase` (Sasieni 1994) are used for fitting a natural cubic spline, which is constrained to be linear outside the completeness region and parameterized using the truncated power basis. The splines fitted using `bspline` or `frencurv`, on the other hand, are unconstrained (hence the extra degrees of freedom corresponding to the external reference points) and parameterized using the B-spline or reference spline basis, respectively. `frencurv` and `bspline` are therefore complementary to the existing programs and do not supersede them.

Saved results

bspline saves in r():

Scalars			
r(xsup)	upper bound of completeness region	r(xinf)	lower bound of completeness region
Macros			
r(nincomp)	number of X-values out of completeness region	r(knots)	final list of knots
r(splist)	*varlist* of splines	r(labfmt)	format used in spline labels
r(type)	storage type of splines (float or double)	r(nknot)	number of knots
r(nspline)	number of splines	r(power)	power (or degree) of splines
r(xvar)	X-variable specified by xvar option		
Matrices			
r(knotv)	row vector of knots		

frencurv saves all of the above results in r(), and also the following:

Macros	
r(refpts)	final list of reference points
Matrices	
r(refv)	row vector of reference points

The result r(nincomp) is the number of values of xvar outside the completeness region of the space of splines defined by the reference splines or B-splines. The number lists r(knots) and r(refpts) are the final lists after any left and right extensions carried out by bspline or frencurv, and the vectors r(knotv) and r(refv) contain the same values in double precision (mainly for programmers). The scalars r(xinf) and r(xsup) are knots, such that the completeness region is $\mathtt{r(xinf)} \leq x < \mathtt{r(xsup)}$.

Acknowledgements

The idea for the name frencurv came from Nick Cox of Durham University, UK, who remarked that the method was like an updated French curve when I described it on Statalist.

References

de Boor, C. 1978. *A practical guide to splines.* New York: Springer-Verlag.

Sasieni, P. 1994. snp7: Natural cubic splines. *Stata Technical Bulletin* 22: 19–22. Reprinted in *Stata Technical Bulletin Reprints*, vol. 4, pp. 171–174.

Unser M., A. Aldroubi, and M. Eden. 1992. On the asymptotic convergence of B-spline wavelets to Gabor functions. *IEEE Transactions on Information Theory* 38: 864–872.

Wold, S. 1971. Analysis of kinetic data by means of spline functions. *Chemica Scripta* 1: 97–102.

——. 1974. Spline functions in data analysis. *Technometrics* 16: 1–11.

Ziegler, Z. 1969. One-sided L_1-approximation by splines of an arbitrary degree. In *Approximations with Special Emphasis on Spline Functions*, ed. I. J. Schoenberg. New York: Academic Press.

sg152 | Listing and interpreting transformed coefficients from certain regression models

J. Scott Long, Indiana University, jslong@indiana.edu
Jeremy Freese, University of Wisconsin-Madison, jfreese@ssc.wisc.edu

Abstract: listcoef is a post-estimation command that facilitates the interpretation of estimated coefficients in regression models for categorical, limited, and count dependent variables. The command provides options to list various transformations of the estimated coefficients, such as percent change, factor change, and standardized coefficients. The user can restrict which coefficients are listed by specifying a variable list or indicating a minimum significance level. The help option provides details on the proper interpretation of coefficients.

Keywords: regression models, list coefficients, standardized coefficients, odds ratios, percent change coefficients, post-estimation.

Introduction

The most effective interpretation of regression models often requires the use of alternative transformations of the canonical parameters that define a model. In Stata, each command for estimating a regression model lists estimates of these fundamental parameters. In some cases, there are options to list transformations of the parameters, such as the or option to list odds ratios in logit-type models or the beta option to list fully standardized coefficients for regress. Other programs, such as mcross introduced by Rogers (1995) conveniently present alternative parameterizations that facilitate interpretation. While Stata is commendably clear and accurate in explaining the meaning of the estimated parameters, in practice it is easy to be confused about proper interpretations. For example, the zip model simultaneously estimates a binary and count model, and it is easy to be confused on the direction of the effects.

This article describes the post-estimation command listcoef which allows users to list estimated coefficients in ways that facilitate interpretation. Users can list coefficients selected by name or significance level, list transformations of the coefficients, and request help to facilitate proper interpretation. The user can quietly estimate their model followed by the listcoef command. Additional information on measures of fit for the model can be obtained with the fitstat command of Long and Freese (2000).

Syntax

listcoef [*varlist*] [, pvalue(#) { factor | percent } std constant matrix help]

Options

pvalue(#) specifies that only coefficients significant at the # significance level or smaller will be printed. If pvalue is not given, coefficients for all levels of significance are listed.

factor specifies that the factor change coefficients should be listed.

percent specifies that percent change coefficients should be listed instead of factor change coefficients.

std specifies that coefficients standardized to a unit variance for the independent and/or dependent variables should be listed. For models with a latent dependent variable, the variance of the latent outcome is estimated.

constant includes the constant(s) in the output.

matrix returns results in r() class matrices. These matrices are defined below.

help includes details for interpreting each coefficient.

Description

listcoef can be used with the regression commands clogit, cnreg, cloglog, gologit, intreg, logistic, logit, mlogit, nbreg, ologit, oprobit, omodel, poisson, probit, regress, zinb, and zip.

listcoef uses several utility ado files. These files are also used in other procedures that the authors are writing and may be useful to other programmers. Only brief descriptions are given here. For more details, type help *command-name* after the programs have been installed.

- _pecats.ado returns the names and values of the categories for models with ordinal, nominal, or binary outcomes. For mlogit it indicates the value of the reference category.
- _pedum returns a scalar indicating if a variable is a dummy variable, defined as having only the values 0, 1, or missing.
- _perhs.ado returns the number of right hand side variables and their names for regression models.
- _pesum.ado computes the means, standard deviations, minima, and maxima for the variables in a regression. Optionally, it determines the medians and whether a variable is binary. Matrices are returned with the first column containing statistics for the dependent variables, with the remaining columns containing information for the independent variables.

Depending on the model estimated and the specified options, listcoef will compute standardized coefficients, factor change in the odds or expected counts, or percent change in the odds or expected counts. The table below lists which options and types of coefficients are available for each estimation command. If an option is the default, it does not need to be specified in the command.

Type	Commands	factor	Option percent	std
Type 1:	regress, probit, cloglog, oprobit, tobit, cnreg, intreg	No	No	Default
Type 2:	logit, logistic, ologit	Default	Yes	Yes
Type 3:	clogit, mlogit, poisson, nbreg, zip, zinb	Default	Yes	No

Example for regress

In the simplest case, one can obtain x-standardized, y-standardized, and fully standardized coefficients after estimating a model with regress. The standard Stata output is

```
. regress job fem phd ment fel art cit
  Source |       SS       df       MS                  Number of obs =     408
---------+------------------------------               F(  6,   401) =   17.78
   Model |  81.0584763     6  13.5097461               Prob > F      =  0.0000
Residual |  304.737915   401  .759944926               R-squared     =  0.2101
---------+------------------------------               Adj R-squared =  0.1983
   Total |  385.796392   407  .947902683               Root MSE      =  .87175
------------------------------------------------------------------------------
     job |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
     fem |  -.1391939   .0902344     -1.543   0.124      -.3165856    .0381977
     phd |   .2726826   .0493183      5.529   0.000       .1757278    .3696375
    ment |   .0011867   .0007012      1.692   0.091      -.0001917    .0025651
     fel |   .2341384   .0948206      2.469   0.014       .0477308    .4205461
     art |   .0228011   .0288843      0.789   0.430      -.0339824    .0795846
     cit |   .0044788   .0019687      2.275   0.023       .0006087     .008349
   _cons |   1.067184   .1661357      6.424   0.000       .7405785     1.39379
------------------------------------------------------------------------------
```

`listcoef` provides additional information:

```
. listcoef, help std constant
regress (N=408): Unstandardized and Standardized Estimates
 Observed SD: .97360294
 SD of Error: .8717482
---------------------------------------------------------------------------
     job |      b         t     P>|t|    bStdX    bStdY   bStdXY      SDofX
---------+-----------------------------------------------------------------
     fem |  -0.13919   -1.543   0.124  -0.0680  -0.1430  -0.0698     0.4883
     phd |   0.27268    5.529   0.000   0.2601   0.2801   0.2671     0.9538
    ment |   0.00119    1.692   0.091   0.0778   0.0012   0.0799    65.5299
     fel |   0.23414    2.469   0.014   0.1139   0.2405   0.1170     0.4866
     art |   0.02280    0.789   0.430   0.0514   0.0234   0.0528     2.2561
     cit |   0.00448    2.275   0.023   0.1481   0.0046   0.1521    33.0599
   _cons |   1.06718    6.424   0.000
---------------------------------------------------------------------------
       b = raw coefficient
       t = t-score for test of b=0
   P>|t| = p-value for t-test
   bStdX = x-standardized coefficient
   bStdY = y-standardized coefficient
  bStdXY = fully standardized coefficient
   SDofX = standard deviation of X
```

Example with logit

The logit model illustrates that `listcoef` can be used to obtain alternative transformations of the basic parameters. We begin by estimating the logit model, which produces the standard output:

```
. logit  lfp k5 k618 age wc hc lwg inc, nolog
Logit estimates                                   Number of obs   =        753
                                                  LR chi2(7)      =     124.48
                                                  Prob > chi2     =     0.0000
Log likelihood = -452.63296                       Pseudo R2       =     0.1209
------------------------------------------------------------------------------
     lfp |      Coef.   Std. Err.       z     P>|z|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
      k5 |  -1.462913   .1970006     -7.426   0.000      -1.849027   -1.076799
    k618 |  -.0645707   .0680008     -0.950   0.342      -.1978499    .0687085
     age |  -.0628706   .0127831     -4.918   0.000      -.0879249   -.0378162
      wc |   .8072738   .2299799      3.510   0.000       .3565215    1.258026
      hc |   .1117336   .2060397      0.542   0.588      -.2920969     .515564
     lwg |   .6046931   .1508176      4.009   0.000       .3090961    .9002901
     inc |  -.0344464   .0082084     -4.196   0.000      -.0505346   -.0183583
   _cons |    3.18214   .6443751      4.938   0.000       1.919188    4.445092
------------------------------------------------------------------------------
```

Most frequently, the logit model is interpreted using factor change coefficients, also known as odds ratios. These are the default option for `listcoef`.

```
. listcoef, help
```

```
logit (N=753): Factor Change in Odds

  Odds of: inLF vs NotInLF

-------------------------------------------------------------------
     lfp |      b         z     P>|z|    e^b    e^bStdX      SDofX
---------+---------------------------------------------------------
      k5 |  -1.46291   -7.426   0.000   0.2316   0.4646     0.5240
    k618 |  -0.06457   -0.950   0.342   0.9375   0.9183     1.3199
     age |  -0.06287   -4.918   0.000   0.9391   0.6020     8.0726
      wc |   0.80727    3.510   0.000   2.2418   1.4381     0.4500
      hc |   0.11173    0.542   0.588   1.1182   1.0561     0.4885
     lwg |   0.60469    4.009   0.000   1.8307   1.4266     0.5876
     inc |  -0.03445   -4.196   0.000   0.9661   0.6698    11.6348
-------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in odds for unit increase in X
 e^bStdX = exp(b*SD of X) = change in odds for SD increase in X
   SDofX = standard deviation of X
```

Alternatively, a user might want to interpret the coefficients in terms of their effect on the latent y^* that underlies the observed variable y. Note that to use `listcoef` to compute standardized coefficients does not require that the model be reestimated.

```
. listcoef, std help

logit (N=753): Unstandardized and Standardized Estimates

 Observed SD: .49562951
   Latent SD: 2.0500391

  Odds of: inLF vs NotInLF

----------------------------------------------------------------------------
     lfp |      b         z     P>|z|    bStdX    bStdY   bStdXY      SDofX
---------+------------------------------------------------------------------
      k5 |  -1.46291   -7.426   0.000  -0.7665  -0.7136  -0.3739     0.5240
    k618 |  -0.06457   -0.950   0.342  -0.0852  -0.0315  -0.0416     1.3199
     age |  -0.06287   -4.918   0.000  -0.5075  -0.0307  -0.2476     8.0726
      wc |   0.80727    3.510   0.000   0.3633   0.3938   0.1772     0.4500
      hc |   0.11173    0.542   0.588   0.0546   0.0545   0.0266     0.4885
     lwg |   0.60469    4.009   0.000   0.3553   0.2950   0.1733     0.5876
     inc |  -0.03445   -4.196   0.000  -0.4008  -0.0168  -0.1955    11.6348
----------------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
   bStdX = x-standardized coefficient
   bStdY = y-standardized coefficient
  bStdXY = fully standardized coefficient
   SDofX = standard deviation of X
```

Example with mlogit

A key to fully interpreting the multinomial logit model is to consider all contrasts among the outcome categories. The standard output from `mlogit` provides contrast between all outcomes and the category specified by `basecategory`. Here we specify the `rrr` option in order to obtain "relative risk ratios" which are also known as factor change coefficients.

```
. mlogit occ white ed exper, basecategory(1) rrr nolog

Multinomial regression                            Number of obs   =        337
                                                  LR chi2(12)     =     166.09
                                                  Prob > chi2     =     0.0000
Log likelihood = -426.80048                       Pseudo R2       =     0.1629

------------------------------------------------------------------------------
     occ |       RRR   Std. Err.       z     P>|z|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
BlueCol  |
   white |   3.443553   2.494631     1.707   0.088       .8324658    14.2445
      ed |   .9053581   .0926011    -0.972   0.331       .7408981   1.106324
   exper |   1.004732   .0174807     0.271   0.786       .9710484   1.039585
---------+--------------------------------------------------------------------
Craft    |
   white |   1.603748   .9691607     0.782   0.434       .4906218   5.242345
      ed |   1.098357   .1071502     0.962   0.336       .9072032   1.329788
   exper |   1.028071   .0171417     1.660   0.097       .9950164   1.062223
---------+--------------------------------------------------------------------
```

```
WhiteCol |
   white |   4.813311    4.34508      1.741   0.082       .8204383    28.23852
      ed |   1.423556   .1669526      3.011   0.003       1.131219    1.791439
   exper |   1.035201   .0194922      1.837   0.066       .9976936    1.074119
---------+--------------------------------------------------------------------
Prof     |
   white |   5.896191   4.451944      2.350   0.019       1.342357    25.89852
      ed |   2.178969   .2497737      6.795   0.000       1.740518     2.72787
   exper |   1.036294   .0186916      1.977   0.048       1.000299    1.073584
------------------------------------------------------------------------------
(Outcome occ==Menial is the comparison group)
```

The standard Stata output allows you to immediately determine the factor change in the odds of each outcome category relative to the comparison group `Menial`. The output does not include comparisons among other outcomes, such as `BlueCol` versus `Craft`. `listcoef` provides all possible comparisons. Since this can generate extensive output, we illustrate two options that limit which coefficients are listed. By specifying the variable `ed`, only coefficients for `ed` are listed. The option `pv(.05)` excludes from printing any contrast that is not significant at the .05 level. Note that the coefficients below correspond to those listed above (for example, the effect of `ed` on the odds of `Prof` versus `Menial` is 2.179 in both sets of output):

```
. listcoef ed, pv(.05)

mlogit (N=337): Factor Change in the Odds of occ when P>|z| < 0.05

Variable: ed (sd= 2.94643)

Odds comparing    |
Group 1 - Group 2 |      b         z     P>|z|     e^b   e^bStdX
------------------+---------------------------------------------
BlueCol -Craft    |  -0.19324   -2.494   0.013   0.8243   0.5659
BlueCol -WhiteCol |  -0.45258   -4.425   0.000   0.6360   0.2636
BlueCol -Prof     |  -0.87828   -8.735   0.000   0.4155   0.0752
Craft   -BlueCol  |   0.19324    2.494   0.013   1.2132   1.7671
Craft   -WhiteCol |  -0.25934   -2.773   0.006   0.7716   0.4657
Craft   -Prof     |  -0.68504   -7.671   0.000   0.5041   0.1329
WhiteCol-BlueCol  |   0.45258    4.425   0.000   1.5724   3.7943
WhiteCol-Craft    |   0.25934    2.773   0.006   1.2961   2.1471
WhiteCol-Prof     |  -0.42569   -4.616   0.000   0.6533   0.2853
WhiteCol-Menial   |   0.35316    3.011   0.003   1.4236   2.8308
Prof    -BlueCol  |   0.87828    8.735   0.000   2.4067  13.3002
Prof    -Craft    |   0.68504    7.671   0.000   1.9838   7.5264
Prof    -WhiteCol |   0.42569    4.616   0.000   1.5307   3.5053
Prof    -Menial   |   0.77885    6.795   0.000   2.1790   9.9228
Menial  -WhiteCol |  -0.35316   -3.011   0.003   0.7025   0.3533
Menial  -Prof     |  -0.77885   -6.795   0.000   0.4589   0.1008
----------------------------------------------------------------
```

Example with zip and zinb

For the `zip` and `zinb` models, the output of `listcoef` makes it much simpler to be sure about the proper interpretation. In the standard output from `zip`, which follows, the direction of effects can be difficult to determine.

```
. zip art fem mar kid5 phd ment, inf(fem mar kid5 phd ment) nolog

Zero-inflated poisson regression                  Number of obs   =        915
                                                  Nonzero obs     =        640
                                                  Zero obs        =        275
Inflation model = logit                           LR chi2(5)      =      78.56
Log likelihood  = -1604.773                       Prob > chi2     =     0.0000

------------------------------------------------------------------------------
     art |      Coef.   Std. Err.       z     P>|z|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
art      |
     fem |  -.2091446   .0634047     -3.299   0.001      -.3334155   -.0848737
     mar |    .103751    .071111      1.459   0.145       -.035624     .243126
    kid5 |  -.1433196   .0474293     -3.022   0.003      -.2362793   -.0503599
     phd |  -.0061662   .0310086     -0.199   0.842       -.066942    .0546096
    ment |   .0180977   .0022948      7.886   0.000       .0135999    .0225955
   _cons |   .6408391   .1213072      5.283   0.000       .4030814    .8785967
---------+--------------------------------------------------------------------
```

```
inflate  |
     fem |   .1097465   .2800813      0.392   0.695      -.4392028    .6586958
     mar |  -.3540108   .3176103     -1.115   0.265      -.9765156    .2684941
    kid5 |   .2171001    .196481      1.105   0.269      -.1679956    .6021958
     phd |   .0012702   .1452639      0.009   0.993      -.2834418    .2859821
    ment |   -.134111   .0452462     -2.964   0.003      -.2227918   -.0454302
   _cons |  -.5770618   .5093853     -1.133   0.257      -1.575439     .421315
------------------------------------------------------------------------------
```

`listcoef` makes the interpretation of coefficients explicit.

```
. listcoef fem phd, help

zip (N=915): Factor Change in Expected Count

 Observed SD: 1.926069

Count Equation: Factor Change in Expected Count for Those Not Always 0

------------------------------------------------------------------
     art |      b         z     P>|z|    e^b    e^bStdX      SDofX
---------+--------------------------------------------------------
     fem |  -0.20914   -3.299   0.001   0.8113   0.9010     0.4987
     phd |   0.10375    1.459   0.145   1.1093   1.0503     0.4732
------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in expected count for unit increase in X
 e^bStdX = exp(b*SD of X) = change in expected count for SD increase in X
   SDofX = standard deviation of X

Binary Equation: Factor Change in Odds of Always 0

------------------------------------------------------------------
 Always0 |      b         z     P>|z|    e^b    e^bStdX      SDofX
---------+--------------------------------------------------------
     fem |   0.10975    0.392   0.695   1.1160   1.0563     0.4987
     phd |  -0.35401   -1.115   0.265   0.7019   0.8458     0.4732
------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in odds for unit increase in X
 e^bStdX = exp(b*SD of X) = change in odds for SD increase in X
   SDofX = standard deviation of X
```

Saved Results

When the `matrix` option is specified, `listcoef` saves in `r(.)` whichever of the following are computed for a particular model. Since saving these matrices can noticeably slow execution for `mlogit`, matrices are only saved when the `matrix` option is specified. The row and column labels of matrices describe each coefficient that is included in each of the following matrices (details on the computation of these coefficients are given in the next section).

`r(b)`	slope or regression coefficients
`r(b_fact)`	factor change coefficients
`r(b_facts)`	x-standardized factor change coefficients
`r(b_p)`	p-values for test of regression coefficients
`r(b_pct)`	percent change coefficients
`r(b_pcts)`	x-standardized percent change coefficients
`r(b_sdx)`	standard deviations for independent variables in finding x- and fully-standardized coefficients
`r(b_std)`	fully standardized coefficients
`r(b_xs)`	x-standardized coefficients
`r(b_ys)`	y or y^*-standardized coefficients
`r(b_z)`	z-values or t-values for regression
`r(cons)`	constant or constants
`r(cons_p)`	p-values or t-values for constant or constants
`r(cons_z)`	z-values or t-values for constant or constants
`r(contrast)`	all contrast from `mlogit`
`r(pvalue)`	a scalar indicating the value specified by the `pvalue` option

For `zip` and `zinb`, the matrices `cons2`, `cons2_p`, `cons2_z`, `b2`, `b2_p`, `b2_z`, `b2_fact`, `b2_facts`, `b2_pct`, and `b2_pcts` provide corresponding results for the binary equation.

In the rest of this insert, we briefly describe each type of coefficient listed by `listcoef`. Full details along with citations to original sources are found in Long (1997). For purposes of illustration, we use an example with only two independent variables.

Standardized coefficients

It is often useful to compute coefficients after some or all of the variables have been standardized to a unit variance. This is particularly useful for the models where the scale of the dependent variable is arbitrary (e.g., `logit`, `probit`). By default, estimation commands express coefficients in the metric of the variables as they are found in the dataset. Standardization can be introduced as follows.

The linear regression model estimated by `regress` can be expressed as

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon \tag{1}$$

The independent variables can be standardized with simple algebra. Let σ_k be the standard deviation of x_k. Then, dividing each x_k by σ_k and multiplying the corresponding β_k by σ_k gives

$$y = \beta_0 + (\sigma_1\beta_1)\,\frac{x_1}{\sigma_1} + (\sigma_2\beta_2)\,\frac{x_2}{\sigma_2} + \varepsilon$$

$\beta_k^{S_x} = \sigma_k\beta_k$ is an *x-standardized coefficient*. For a continuous variable, $\beta_k^{S_x}$ can be interpreted as follows. For a standard deviation increase in x_k, y is expected to change by $\beta_k^{S_x}$ units, holding all other variables constant.

To standardize for the dependent variable, let σ_y be the standard deviation of y. We can standardize y by dividing (1) by σ_y, thus giving

$$\frac{y}{\sigma_y} = \frac{\beta_0}{\sigma_y} + \frac{\beta_1}{\sigma_y}x_1 + \frac{\beta_2}{\sigma_y}x_2 + \frac{\varepsilon}{\sigma_y}$$

Then $\beta_k^{S_y} = \beta_k/\sigma_y$ is a *y-standardized coefficient* that can be interpreted as follows. For a unit increase in x_k, y is expected to change by $\beta_k^{S_y}$ standard deviations, holding all other variables constant.

For a dummy variable, the interpretation would be as follows. Having characteristic x_k (as opposed to not having the characteristic) results in an expected change in y of $\beta_k^{S_y}$ standard deviations, holding all other variables constant.

It is also possible to standardize both y and the x's as in

$$\frac{y}{\sigma_y} = \frac{\beta_0}{\sigma_y} + \left(\frac{\sigma_1\beta_1}{\sigma_y}\right)\frac{x_1}{\sigma_1} + \left(\frac{\sigma_2\beta_2}{\sigma_y}\right)\frac{x_2}{\sigma_2} + \frac{\varepsilon}{\sigma_y}$$

Then, $\beta_k^S = (\sigma_k\beta_k)/\sigma_y$ is a *fully standardized coefficient* that can be interpreted as follows. For a standard deviation increase in x_k, y is expected to change by β_k^S standard deviations, holding all other variables constant.

A variety of other models (`logit`, `tobit`, `cnreg`, `intreg`, `probit`, `ologit`, `oprobit`) can be written as

$$y^* = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon \tag{2}$$

where y^* is a latent variable.

In models with a latent dependent variable, (2) can be divided by σ_{y^*}. To estimate the variance of the latent variable, the quadratic form is used.

$$\widehat{\text{Var}(y^*)} = \widehat{\beta}'\,\widehat{\text{Var}}(x)\widehat{\beta} + \text{Var}(\epsilon)$$

where $\widehat{\beta}$ is a vector of estimated coefficients and $\widehat{\text{Var}}(x)$ is the covariance matrix for the x's computed from the observed data. In some models such as tobit, $\text{Var}(\epsilon)$ is estimated from the data. In probit models, $\text{Var}(\epsilon) = 1$ by assumption and $\text{Var}(\epsilon) = \pi^2/3$ in logit models.

Factor and percent change

In logit-based models and models for counts, coefficients can be expressed either as a factor or multiplicative change in the odds or the expected count, or as a percentage change. In `logit` the `or` option computes the factor change in the odds, referred to as odds ratios. In `mlogit`, the `rrr` option computes the relative risk ratio which is also a factor change in the odds.

If $\Omega(x, x_k)$ is the odds of some outcome (for example, working versus not working) for a given set of independent variables, the transformation

$$e^{\beta_k} = \frac{\Omega(x, x_k + 1)}{\Omega(x, x_k)}$$

is the factor or multiplicative change in the predicted odds when x_k changes by one unit. Thus we have the *factor change*: for a unit change in x_k, the odds are expected to change by a factor of $\exp(\beta_k)$, holding all other variables constant.

The effect of a change of a standard deviation change s_k in x_k will equal $\exp{(\beta_k \times s_k)}$, thus giving rise to the *standardized factor change*: for a standard deviation change in x_k, the odds are expected to change by a factor of $\exp(\beta_k \times s_k)$, holding all other variables constant.

For the binary logit model estimated by `logit` or `logistic`, the odds are for outcome "not-0" versus outcome 0'. Often the dependent variable is coded as 1 and 0 so it becomes the odds of a 1 compared to a 0. The coefficients in the ordinal logit model estimated by `ologit` can also be interpreted in terms of factor change in the odds. In this model, the odds are for "outcomes greater than some value" compared to "outcomes less than some value", for example, the odds for categories 3 or 4 compared to categories 1 and 2.

Instead of a multiplicative or factor change in the outcome, some people prefer the *percent change* given by

$$100\,[\,\exp{(\beta_k \times \delta)} - 1]$$

which is listed by `listcoef` with the `percent` option.

Count models

The `poisson` and `nbreg` models are loglinear in the mean of the expected count. For example in the Poisson model,

$$E(y \mid x) = e^{\beta_0}\, e^{\beta_1 x_1} e^{\beta_2 x_2}$$

Accordingly, $\exp{(\beta_k)}$ can be interpreted as follows. For a unit change in x_k, the expected count changes by a factor of $\exp(\beta_k)$, holding all other variables constant.

For a standard deviation change s_k in x_k, the *factor change coefficient* $\exp{(\beta_k \times s_k)}$ can be interpreted as follows. For a standard deviation change in x_k, the expected count changes by a factor of $\exp(\beta_k \times s_k)$, holding all other variables constant.

Alternatively, the percentage change in the expected count for a δ unit change in x_k, holding other variables constant, can be computed as

$$100 \times [\,\exp(\beta_k \times \delta) - 1\,]$$

Zero-inflated models

The zero-inflated models `zip` and `zinb` combine a binary model predicting those who always have 0 outcomes and those who might not have zeros and a count model that applies to those who could have non-zero outcomes. If the binary portion of the model is assumed to be a logit, interpretation of the binary portion can be made in terms of the odds of always being 0 versus not always being 0. The count portion can be interpreted as a standard count model. `listcoef` transforms coefficients accordingly and the `help` option provides information on the correct interpretation.

Alternative contrasts for mlogit

`mlogit` estimates the multinomial logit in which the estimated coefficients are for the comparison of a given outcome to a base category. For complete interpretation it is useful to examine the coefficients for all possible comparisons, including comparisons between categories in which neither outcome is the base category. While this could be done by specifying a different `basecategory`, all possible comparisons are computed by `listcoef`. Since this can involve a large number of coefficients when there are more than three dependent categories, the `pvalue` options can be useful for finding the most important effects.

Acknowledgment

We thank David Drukker at Stata Corporation for his suggestions.

References

Long, J. S. 1997. *Regression Models for Categorical and Limited Dependent Variables*. Thousand Oaks, CA: Sage.

Long, J. S. and J. Freese. 2000. sg145: Scalar measures of fit for regression models. *Stata Technical Bulletin* 56: 34–40.

Rogers, W. H. 1995. sqv10: Expanded multinomial comparisons. *Stata Technical Bulletin* 23: 26–28. Reprinted in *Stata Technical Bulletin Reprints*, vol. 4, pp. 181–183.

snp15.1	Update to somersd

Roger Newson, Guy's, King's and St Thomas' School of Medicine, London, UK, roger.newson@kcl.ac.uk

Abstract: `somersd` calculates confidence intervals for rank-order statistics. It has been improved, streamlined, debugged, and intensively certified.

Keywords: Somers' D, Kendall's tau, rank correlation, confidence intervals, nonparametric methods.

Syntax

`somersd` [*varlist*] [*weight*] [`if` *exp*] [`in` *range*] [`, cluster(`*varname*`) level(`*#*`) taua tdist`

`transf(`*transformation_name*`) cimatrix(`*new_matrix*`)`]

where *transformation_name* is one of

`iden` | `z` | `asin` | `rho` | `zrho`

`fweight`s, `iweight`s, and `pweight`s are allowed.

New options

`cimatrix(`*new_matrix*`)` specifies an output matrix to be created, containing estimates and confidence limits for the untransformed Somers' D, Kendall's τ_a or Greiner's ρ parameters. If `transf()` is specified, then the confidence limits will be asymmetric and based on symmetric confidence limits for the transformed parameters. This option (like `level`) may be used in replay mode as well as in non-replay mode.

New saved results

`somersd` now saves additionally the name of the program called by `predict` in the macro `e(predict)`.

Remarks

`somersd` was introduced in Newson (2000). The program calculates confidence intervals for the rank order statistics Somers' D and Kendall's τ_a for the first variable of *varlist* as a predictor of each of the other variables in *varlist*, with estimates and jackknife covariances saved as estimation results. The new version contains the following improvements:

1. The new option `cimatrix` has been added (mostly for programmers).

2. The program `somers_p` has been added as the `predict` program for `somersd`, and it warns the user that `predict` should not be used after `somersd`.

3. `somersd` has been streamlined. If `cluster()` is not specified, then processing time is now quadratically dependent on the number of distinct value combinations in `varlist`, instead of being quadratically dependent on the number of observations as before. This makes a vast difference to the time taken to process discrete variables in data sets with thousands of observations.

4. A bug has been corrected, which formerly caused incorrect output when the `taua` option was used with unequal `fweight`s. (This bug was not present in the earlier version of `somersd` circulated via the Ideas list, and there was no excuse for me to allow it to creep in when upgrading `somersd` for the STB.)

5. The certification script used to certify `somersd` is now much more comprehensive than before, ruling out the above bug and a large range of others. (See the online `help cscript`.) Amongst other checks, it checks its jackknife confidence intervals for Kendall's τ_a with those produced by `ktau` and `jknife` (Gould 1995). The latter programs produce the same confidence limits as `somersd, taua tdist` in the most simple case, without weights, clustering or transformations. However, `ktau` and `jknife` take much longer, requiring a time *cubically* dependent on the number of observations.

Acknowledgments

I would like to thank Bill Gould of Stata Corporation for suggesting the `somers_p` program, and Bill Gould and Ken Higbee of Stata Corporation for a great deal of very helpful advice on designing certification scripts.

References

Gould, W. 1995. sg34: Jackknife estimation. *Stata Technical Bulletin* 24: 25–29. Reprinted in *Stata Technical Bulletin Reprints*, vol. 4, pp. 165–170.

Newson, R. 2000. snp15: `somersd`—Confidence limits for nonparametric statistics and their differences. *Stata Technical Bulletin* 55: 47–55.

sts15	Tests for stationarity of a time series

Christopher F. Baum, Boston College, baum@bc.edu

Abstract: Implements the Elliott–Rothenberg–Stock (1996) DF-GLS test and the Kwiatkowski–Phillips–Schmidt–Shin (1992) KPSS tests for stationarity of a time series. The DF-GLS test is an improved version of the augmented Dickey–Fuller test. The KPSS test has a null hypothesis of stationarity and may be employed in conjunction with the DF-GLS test to detect long memory (fractional integration).

Keywords: stationarity, unit root, time series.

Syntax

`dfgls` *varname* [`if` *exp*] [`in` *range*] [, `maxlag(`*#*`)` `notrend` `ers`]

`kpss` *varname* [`if` *exp*] [`in` *range*] [, `maxlag(`*#*`)` `notrend`]

Both tests are for use with time series data; you must `tsset` your data before using these tests; see [R] **tsset**. *varname* may contain time series operators; see [U] **14.4.3 Time series varlists**.

Options

`maxlag(`*#*`)` specifies the maximum lag order to be considered. The test statistics will be calculated for each lag up to the maximum lag order (which may be zero). If not specified, the maximum lag order for the test is by default calculated from the sample size using a rule provided by Schwert (1989) using $c = 12$ and $d = 4$ in his terminology. Whether the maximum lag is explicitly specified or computed by default, the sample size is held constant over lags at the maximum available sample.

`notrend` specifies that no trend term should be included in the model. The critical values reported differ in the absence of a trend term.

`ERS` (`dfgls` only) specifies that the ERS (and Dickey–Fuller) values are to be used for all levels of significance (eschewing the response surface estimates).

Description

`dfgls` performs the Elliott–Rothenberg–Stock (ERS, 1996) efficient test for an autoregressive unit root. This test is similar to an (augmented) Dickey–Fuller t test, as performed by `dfuller`, but has the best overall performance in terms of small sample size and power, dominating the ordinary Dickey–Fuller test. The `dfgls` test "has substantially improved power when an unknown mean or trend is present" (ERS, 813).

`dfgls` applies a generalized least squares (GLS) detrending (demeaning) step to the *varname*

$$y_t^d = y_t - \widehat{\beta}' z_t$$

For detrending, $z_t = (1, t)'$ and $\widehat{\beta}_0$, $\widehat{\beta}_1$ are calculated by regressing

$$[y_1, (1 - \bar{\alpha}L)\, y_2, \ldots, (1 - \bar{\alpha}L)\, y_T]$$

onto

$$[z_1, (1 - \bar{\alpha}L)\, z_2, ..., (1 - \bar{\alpha}L)\, z_T]$$

where $\bar{\alpha} = 1 + \bar{c}/T$ with $\bar{c} = -13.5$, and L is the lag operator. For demeaning, $z_t = (1)'$ and the same regression is run with $\bar{c} = -7.0$. The values of $\bar{c}$ are chosen so that "the test achieves the power envelope against stationary alternatives (is asymptotically MPI (most powerful invariant)) at 50 percent power" (Stock 1994, 2769; emphasis added). The augmented Dickey–Fuller regression is then computed using the y_t^d series

$$\Delta y_t^d = \alpha + \gamma t + \rho y_{t-1}^d + \sum_{i=1}^{m} \delta_i \Delta y_{t-i}^d + \epsilon_t$$

where m=`maxlag`. The `notrend` option suppresses the time trend in this regression.

Approximate 5% and 10% critical values, by default, are calculated from the response surface estimates of Table 1, Cheung and Lai (1995, 413), which take both the sample size and the lag specification into account. Approximate 1% critical values for the GLS detrended test are interpolated from Table 1 of ERS (page 825). Approximate 1% critical values for the GLS demeaned test are identical to those applicable to the no-constant, no-trend Dickey–Fuller test and are computed using the **dfuller** code. The **ERS** option specifies that the ERS (and Dickey–Fuller) values are to be used for all levels of significance (eschewing the response surface estimates).

If the maximum lag order exceeds one, the optimal lag order is calculated by the Ng and Perron (1995) sequential t test on the highest order lag coefficient, stopping when that coefficient's p-value is less than 0.10. The lag minimizing the Schwarz criterion (SC, or BIC) is printed with its minimized value.

kpss performs the Kwiatkowski–Phillips–Schmidt–Shin test introduced in Kwiatkowski et al. (1992) for stationarity of a time series. This test differs from those in common use (such as **dfuller** and **pperron**) by having a null hypothesis of stationarity. The test may be conducted under the null hypothesis of either trend stationarity (the default) or level stationarity. Inference from this test is complementary to that derived from those based on the Dickey–Fuller distribution (such as **dfgls**, **dfuller** and **pperron**). The KPSS test is often used in conjunction with those tests to investigate the possibility that a series is fractionally integrated; that is, neither $I(1)$ nor $I(0)$; see Lee and Schmidt (1996).

The series is detrended (demeaned) by regressing y on $z_t = (1,t)^{'}$ $\left(z_t = (1)'\right)$, yielding residuals e_t. Let the partial sum series of e_t be s_t. Then the zero-order KPSS statistic $k_0 = T^{-2}\sum_{t=1}^{T} s_t^2 / T^{-1}\sum_{t=1}^{T} e_t^2$. For **maxlag** > 0, the denominator is computed as the Newey–West estimate of the long run variance of the series; see [R] **newey**.

Approximate critical values for the KPSS test are taken from Kwiatkowski et al. (1992).

Examples

Data from Terence Mills' *Econometric Analysis of Financial Time Series* on the UK FTA All Share Index of stock prices (**ftap**) and stock returns (**ftaret**) are analyzed.

```
. use http://fmwww.bc.edu/ec-p/data/Mills2d/fta.dta

. tsset
        time variable:  month, 1965m1 to 1995m12

. dfgls ftap

Number of obs =   355
Maxlag = 16 chosen by Schwert criterion


                      Test       1% Critical       5% Critical      10% Critical
                 Statistic           Value             Value             Value
--------------------------------------------------------------------------------
DF-GLS(tau)[16]    -0.068          -3.480            -2.818            -2.536
DF-GLS(tau)[15]    -0.155          -3.480            -2.824            -2.542
DF-GLS(tau)[14]    -0.046          -3.480            -2.829            -2.547
DF-GLS(tau)[13]    -0.234          -3.480            -2.835            -2.552
DF-GLS(tau)[12]    -0.131          -3.480            -2.840            -2.557
DF-GLS(tau)[11]    -0.196          -3.480            -2.846            -2.562
DF-GLS(tau)[10]    -0.251          -3.480            -2.851            -2.566
DF-GLS(tau)[9]     -0.173          -3.480            -2.856            -2.571
DF-GLS(tau)[8]     -0.107          -3.480            -2.861            -2.575
DF-GLS(tau)[7]     -0.361          -3.480            -2.865            -2.580
DF-GLS(tau)[6]     -0.391          -3.480            -2.870            -2.584
DF-GLS(tau)[5]     -0.476          -3.480            -2.874            -2.588
DF-GLS(tau)[4]     -0.524          -3.480            -2.879            -2.592
DF-GLS(tau)[3]     -0.484          -3.480            -2.883            -2.595
DF-GLS(tau)[2]     -0.507          -3.480            -2.887            -2.599
DF-GLS(tau)[1]     -0.789          -3.480            -2.891            -2.602

Opt Lag (Ng-Perron sequential t) = 15 with RMSE  35.59803

Min SC =  7.275482 at lag  2 with RMSE    37.0745

. kpss ftap

KPSS test for ftap

Maxlag = 16 chosen by Schwert criterion

Critical values for H0: ftap is trend stationary

10%: 0.119  5% : 0.146  2.5%: 0.176  1% : 0.216

Lag order    Test statistic
    0       7.90141
    1       4.18402
    2       2.86036
    3       2.18027
    4       1.76579
```

```
    5       1.48676
    6        1.2861
    7       1.13475
    8       1.01642
    9       .921225
   10        .84288
   11       .777242
   12       .721428
   13       .673349
   14       .631492
   15       .594708
   16       .562121
. dfgls ftaret
Number of obs =   355
Maxlag = 16 chosen by Schwert criterion
                   Test        1% Critical       5% Critical      10% Critical
               Statistic          Value             Value             Value
------------------------------------------------------------------------------
DF-GLS(tau)[16]    -4.161           -3.480            -2.818            -2.536
DF-GLS(tau)[15]    -4.119           -3.480            -2.824            -2.542
DF-GLS(tau)[14]    -4.413           -3.480            -2.829            -2.547
DF-GLS(tau)[13]    -4.733           -3.480            -2.835            -2.552
DF-GLS(tau)[12]    -4.663           -3.480            -2.840            -2.557
DF-GLS(tau)[11]    -4.392           -3.480            -2.846            -2.562
DF-GLS(tau)[10]    -4.653           -3.480            -2.851            -2.566
DF-GLS(tau)[9]     -4.795           -3.480            -2.856            -2.571
DF-GLS(tau)[8]     -4.931           -3.480            -2.861            -2.575
DF-GLS(tau)[7]     -6.006           -3.480            -2.865            -2.580
DF-GLS(tau)[6]     -6.203           -3.480            -2.870            -2.584
DF-GLS(tau)[5]     -6.911           -3.480            -2.874            -2.588
DF-GLS(tau)[4]     -7.614           -3.480            -2.879            -2.592
DF-GLS(tau)[3]     -7.769           -3.480            -2.883            -2.595
DF-GLS(tau)[2]     -9.176           -3.480            -2.887            -2.599
DF-GLS(tau)[1]    -13.075           -3.480            -2.891            -2.602
Opt Lag (Ng-Perron sequential t) =  8 with RMSE  .0593867
Min SC =  -5.566828 at lag  2 with RMSE  .0603119
. dfgls ftaret,notrend
Number of obs =   355
Maxlag = 16 chosen by Schwert criterion
                   Test        1% Critical       5% Critical      10% Critical
               Statistic          Value             Value             Value
------------------------------------------------------------------------------
DF-GLS(mu)[16]     -3.165           -2.580            -1.952            -1.637
DF-GLS(mu)[15]     -3.161           -2.580            -1.955            -1.640
DF-GLS(mu)[14]     -3.430           -2.580            -1.958            -1.643
DF-GLS(mu)[13]     -3.725           -2.580            -1.962            -1.646
DF-GLS(mu)[12]     -3.711           -2.580            -1.965            -1.649
DF-GLS(mu)[11]     -3.528           -2.580            -1.968            -1.652
DF-GLS(mu)[10]     -3.776           -2.580            -1.971            -1.655
DF-GLS(mu)[9]      -3.933           -2.580            -1.974            -1.658
DF-GLS(mu)[8]      -4.087           -2.580            -1.977            -1.660
DF-GLS(mu)[7]      -5.039           -2.580            -1.980            -1.663
DF-GLS(mu)[6]      -5.278           -2.580            -1.982            -1.665
DF-GLS(mu)[5]      -5.966           -2.580            -1.985            -1.668
DF-GLS(mu)[4]      -6.679           -2.580            -1.988            -1.670
DF-GLS(mu)[3]      -6.928           -2.580            -1.990            -1.672
DF-GLS(mu)[2]      -8.312           -2.580            -1.993            -1.675
DF-GLS(mu)[1]     -12.060           -2.580            -1.995            -1.677
Opt Lag (Ng-Perron sequential t) =  8 with RMSE  .0600067
Min SC = -5.53158 at lag  2 with RMSE  .0613843
```

Both tests indicate that `ftap` appears to be nonstationary. `ftaret` appears to be both trend and level stationary.

Saved Results

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

`r(N)`	number of observations
`r(optlag)`	optimal lag order
`r(sc`*n*`)`	Schwarz criterion at lag *n*
`r(rmse`*n*`)`	root mean square error at lag *n*
`r(dft`*n*`)`	DF-GLS statistic at lag *n*

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

`r(N)`	number of observations
`r(dft`*n*`)`	KPSS statistic at lag *n*

Acknowledgments

I acknowledge useful conversations with Serena Ng, James Stock, and Vince Wiggins. The KPSS code was adapted from John Barkoulas' RATS code for that test. Thanks also to Richard Sperling for tracking down a discrepancy between published work and the `dfgls` output and alerting me to the Cheung and Lai estimates. Any remaining errors are my own.

References

Cheung, Y. W. and K.-S. Lai. 1995. Lag order and critical values of a modified Dickey–Fuller test. *Oxford Bulletin of Economics and Statistics* 57: 411–419.

Elliott, G., T. J. Rothenberg, and J. H. Stock. 1996. Efficient tests for an autoregressive unit root. *Econometrica* 64: 813–836.

Kwiatkowski, D., P. C. Phillips, P. Schmidt, and Y. Shin. 1992. Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of Econometrics* 54: 159–178.

Lee, D. and P. Schmidt. 1996. On the power of the KPSS test of stationarity against fractionally-integrated alternatives. *Journal of Econometrics* 73: 285–302.

Ng, S. and P. Perron. 1995. Unit root tests in ARMA models with data-dependent methods for the selection of the truncation lag. *Journal of the American Statistical Association* 90: 268–281.

Schwert, G. W. 1989. Tests for unit roots: A Monte Carlo investigation. *Journal of Business and Economic Statistics* 7: 147–160.

Stock, J. H. 1994. Unit roots, structural breaks and trends. In *Handbook of Econometrics IV*, ed. R. F. Engle and D. L. McFadden. Amsterdam: Elsevier.

sts16	Tests for long memory in a time series

Christopher F. Baum, Boston College, baum@bc.edu
Vince Wiggins, Stata Corporation, vwiggins@stata.com

Abstract: Implements the Geweke/Porter-Hudak log periodogram estimator (1983), the Phillips modified log periodogram estimator (1999b) and the Robinson log periodogram estimator (1995) for the diagnosis of long memory, or fractional integration, in a time series. The Robinson estimator may be applied to a set of time series.

Keywords: fractional integration, long memory, stationarity, time series.

Syntax

`gphudak` *varname* [`if` *exp*] [`in` *range*] [`, powers(`*numlist*`)`]

`modlpr` *varname* [`if` *exp*] [`in` *range*] [`, powers(`*numlist*`) notrend`]

`roblpr` *varlist* [`if` *exp*] [`in` *range*] [`, powers(`*numlist*`) l(`*#*`) j(`*#*`) constraints(`*numlist*`)`]

These tests are for use with time series data; you must `tsset` your data before using these tests; see [R] **tsset**. *varname* or *varlist* may contain time series operators; see [U] **14.4.3 Time-series varlists**.

Options

`powers(`*numlist*`)` indirectly specifies the number of ordinates to be included in the regression. A number of ordinates equal to the integer part of T raised to the `powers(`*numlist*`)` will be used. Powers ranging from 0.50 to 0.75 are commonly employed for `gphudak` and `modlpr`. These routines use the default power of 0.5. `roblpr` uses the default power of 0.9. For `roblpr`, multiple powers may only be specified if a single variable appears in *varlist*.

`notrend` specifies that detrending is not to be applied by `modlpr`. By default, a linear trend will be removed from the series.

`l(#)` specifies the number of initial ordinates to be removed from the regression for `roblpr`. Some researchers have found that such exclusion improves the properties of tests based on log-periodogram regressions. The default value of `l` is zero.

`j(#)` specifies that the log periodogram employed in `roblpr` is to be computed as an average of adjacent ordinates. The default value of `j` is 1, so that no averaging is performed. If `j` is 2, the number of ordinates is halved; with a `j` of 3, divided by three, and so on. When `j` is greater than 1, the value of `powers` should be set large enough so that the averaged ordinates are sufficient in number.

`constraints(`*numlist*`)` specifies the constraint numbers of the linear constraints to be applied during estimation in `roblpr`. The default is to perform unconstrained estimation. This option allows the imposition of linear constraints prior to estimation of the pooled coefficient vector. For instance, if *varlist* contains prices, dividends, and returns, and your prior (or previous findings) states that prices' and dividends' order of integration is indistinguishable, one might impose that constraint to improve the power of the F test provided by `roblpr`. You would specify the constraints prior to the `roblpr` command and then provide the list of constraints in the constraints option to `roblpr`.

Technical note on constraints. When constraints are imposed it is difficult to identify the number of numerator degrees of freedom in the test for equality of d coefficients reported at the bottom of `roblpr`'s output. Since constraints can be of any general form and it is possible to specify constraints that are not unique, `roblpr` determines the degrees of freedom from the rank of the matrix used to compute the Wald statistic. Determining that matrix rank from a numerical standpoint can be problematic, in which case `roblpr` may overstate the number of constraints being tested and thereby incorrectly compute the numerator degrees of freedom for the test. This rarely has a meaningful impact on the statistical test, but you may wish to test only the unconstrained coefficients if the computed degrees of freedom are wrong.

For example, after the final example below, we could perform the test by typing `test ftap == ftaret`. In this case, the degrees of freedom were correct, so we needn't have gone to the trouble.

Description

The model of an autoregressive fractionally integrated moving average process of a time series of order $(p,\, d,\, q)$, denoted by ARFIMA$(p,\, d,\, q)$, with mean μ, may be written using operator notation as

$$\Phi(L)(1-L)^d\,(y_t-\mu)=\Theta(L)\epsilon_t,\ \epsilon_t\sim i.i.d.(0,\sigma^2_\epsilon) \tag{1}$$

where L is the backward-shift operator,

$$\Phi(L)=1-\phi_1 L-\cdots-\phi_p L^p$$

$\Theta(L)=1+\vartheta_1 L+\cdots+\vartheta_q L^q$, and $(1-L)^d$ is the fractional differencing operator defined by

$$(1-L)^d=\sum_{k=0}^{\infty}\frac{\Gamma(k-d)L^k}{\Gamma(-d)\Gamma(k+1)} \tag{2}$$

with $\Gamma(\cdot)$ denoting the gamma (generalized factorial) function. The parameter d is allowed to assume any real value. The arbitrary restriction of d to integer values gives rise to the standard autoregressive integrated moving average (ARIMA) model. The stochastic process y_t is both stationary and invertible if all roots of $\Phi(L)$ and $\Theta(L)$ lie outside the unit circle and $|d|<0.5$. The process is nonstationary for $d\geq 0.5$, as it possesses infinite variance; for example, see Granger and Joyeux (1980).

Assuming that $d\in[0,0.5)$, Hosking (1981) showed that the autocorrelation function, $\rho(\cdot)$, of an ARFIMA process is proportional to k^{2d-1} as $k\to\infty$. Consequently, the autocorrelations of the ARFIMA process decay hyperbolically to zero as $k\to\infty$ in contrast to the faster, geometric decay of a stationary ARMA process. For $d\in(0,0.5)$, $\sum_{j=-n}^{n}|\rho(j)|$ diverges as $n\to\infty$, and the ARFIMA process is said to exhibit long memory, or long-range positive dependence. The process is said to exhibit intermediate memory (anti-persistence), or long-range negative dependence, for $d\in(-0.5,0)$. The process exhibits short memory for $d=0$, corresponding to stationary and invertible ARMA modeling. For $d\in[0.5,1)$ the process is mean reverting, even though it is not covariance stationary, as there is no long-run impact of an innovation on future values of the process.

If a series exhibits long memory, it is neither stationary ($I(0)$) nor is it a unit root ($I(1)$) process; it is an $I(d)$ process, with d a real number. A series exhibiting long memory, or persistence, has an autocorrelation function that damps hyperbolically, more slowly than the geometric damping exhibited by "short memory" (ARMA) processes. Thus, it may be predictable at long horizons. Long memory models originated in hydrology and have been widely applied in economics and finance. An excellent survey of long memory models is given by Baillie (1996).

There are two approaches to the estimation of an ARFIMA $(p,\, d,\, q)$ model: exact maximum likelihood estimation, as proposed by Sowell (1992), and semiparametric approaches, as described in this insert. Sowell's approach requires specification of the p and q values, and estimation of the full ARFIMA model conditional on those choices. This involves all the attendant difficulties of choosing an appropriate ARMA specification, as well as a formidable computational task for each combination of p and q to be evaluated. The methods described here assume that the short memory or ARMA components of the time series are relatively unimportant, so that the long memory parameter d may be estimated without fully specifying the data-generating process. These methods are thus described as semiparametric.

`gphudak` performs the Geweke and Porter-Hudak (GPH 1983) semiparametric log periodogram regression, often described as the "GPH test," for long memory (fractional integration) in a time series. The GPH method uses nonparametric methods—a spectral regression estimator—to evaluate d without explicit specification of the ARMA parameters of the series. The series is usually differenced so that the resulting d estimate will fall in the $[-0.5, 0.5]$ interval.

Geweke and Porter-Hudak (1983) proposed a semiparametric procedure to obtain an estimate of the memory parameter d of a fractionally integrated process X_t in a model of the form

$$(1-L)^d X_t = \epsilon_t, \tag{3}$$

where ϵ_t is stationary with zero mean and continuous spectral density $f_\epsilon(\lambda) > 0$. The estimate $\widehat{d}$ is obtained from the application of ordinary least squares to

$$\log\left(I_x(\lambda_s)\right) = \widehat{c} - \widehat{d}\log\left|1-e^{i\lambda_s}\right|^2 + \text{residual} \tag{4}$$

computed over the fundamental frequencies $\{\lambda_s = 2\pi s/n, s = 1, \ldots, m < n\}$. We define

$$\omega_x(\lambda_s) = \frac{1}{\sqrt{2\pi n}}\sum_{t=1}^{n} X_t e^{it\lambda_s}$$

as the discrete Fourier transform (DFT) of the time series X_t, $I_x(\lambda_s) = \omega_x(\lambda_s)\,\omega_x(\lambda_s)^*$ as the periodogram, and $x_s = \log\left|1-e^{i\lambda_s}\right|$. Ordinary least squares on (4) yields

$$\widehat{d} = \frac{\sum_{s=1}^{m} x_s \log I_x(\lambda_s)}{2\sum_{s=1}^{m} x_s^2} \tag{5}$$

Various authors have proposed methods for the choice of m, the number of Fourier frequencies included in the regression. The regression slope estimate is an estimate of the slope of the series' power spectrum in the vicinity of the zero frequency; if too few ordinates are included, the slope is calculated from a small sample. If too many are included, medium and high-frequency components of the spectrum will contaminate the estimate. A choice of $\sqrt{T}$ or 0.5 for `power` is often employed. To evaluate the robustness of the GPH estimate, a range of power values (from 0.40 to 0.75) is commonly calculated as well. Two estimates of the d coefficient's standard error are commonly employed: the regression standard error, giving rise to a standard t test, and an asymptotic standard error, based upon the theoretical variance of the log periodogram of $\pi^2/6$. The statistic based upon that standard error has a standard normal distribution under the null.

`modlpr` computes a modified form of the GPH estimate of the long memory parameter, d, of a time series, proposed by Phillips (1999a, 1999b). Phillips (1999a) points out that the prior literature on this semiparametric approach does not address the case of $d = 1$, or a unit root, in (3), despite the broad interest in determining whether a series exhibits unit-root behavior or long memory behavior, and his work showing that the $\widehat{d}$ estimate of (5) is inconsistent when $d > 1$, with $\widehat{d}$ exhibiting asymptotic bias toward unity. This weakness of the GPH estimator is solved by Phillips' modified log periodogram regression estimator, in which the dependent variable is modified to reflect the distribution of d under the null hypothesis that $d = 1$. The estimator gives rise to a test statistic for $d = 1$ which is a standard normal variate under the null. Phillips suggests that deterministic trends should be removed from the series before application of the estimator. Accordingly, the routine will automatically remove a linear trend from the series. This may be suppressed with the `notrend` option. The comments above regarding `power` apply equally to `modlpr`.

The Phillips (1999b) modification of the GPH estimator is based on an exact representation of the DFT in the unit root case. The modification expresses

$$\omega_x(\lambda_s) = \frac{\omega_u(\lambda_s)}{1-e^{i\lambda_s}} - \frac{e^{i\lambda_s}}{1-e^{i\lambda_s}}\frac{X_n}{\sqrt{2\pi n}}$$

and the modified DFT as

$$\upsilon_x(\lambda_s) = \omega_x(\lambda_s) + \frac{e^{i\lambda_s}}{1-e^{i\lambda_s}}\frac{X_n}{\sqrt{2\pi n}}$$

with associated periodogram ordinates $I_v(\lambda_s) = \upsilon_x(\lambda_s)\,\upsilon_x(\lambda_s)^*$ (Phillips 1999b, 9). He notes that both $\upsilon_x(\lambda_s)$ and, thus, $I_v(\lambda_s)$ are observable functions of the data. The log-periodogram regression is now the regression of $\log I_v(\lambda_s)$ on $a_s = \log\left|1 - e^{i\lambda_s}\right|$. Defining $\bar{a} = m^{-1}\sum_{s=1}^{m} a_s$ and $x_s = a_s - \bar{a}$, the modified estimate of the long-memory parameter becomes

$$\tilde{d} = \frac{\sum_{s=1}^{m} x_s \log I_\nu(\lambda_s)}{2\sum_{s=1}^{m} x_s^2} \tag{6}$$

Phillips proves that, with appropriate assumptions on the distribution of ϵ_t, the distribution of $\tilde{d}$ follows

$$\sqrt{m}\left(\tilde{d} - d\right) \rightarrow N\left(0, \frac{\pi^2}{24}\right) \tag{7}$$

in distribution, so $\tilde{d}$ has the same limiting distribution at $d = 1$ as does the GPH estimator in the stationary case so $\tilde{d}$ is consistent for values of d around unity. A semiparametric test statistic for a unit root against a fractional alternative is then based upon the statistic (Phillips 1999a, 10)

$$z_d = \frac{\sqrt{m}\left(\tilde{d} - 1\right)}{\pi/24} \tag{8}$$

with critical values from the standard normal distribution. This test is consistent against both $d < 1$ and $d > 1$ fractional alternatives.

`roblpr` computes the Robinson (1995) multivariate semiparametric estimate of the long memory (fractional integration) parameters, $d(g)$, of a set of G time series, $y(g)$, $g = 1, G$ with $G \geq 1$. When applied to a set of time series, the $d(g)$ parameter for each series is estimated from a single log-periodogram regression which allows the intercept and slope to differ for each series. One of the innovations of Robinson's estimator is that it is not restricted to using a small fraction of the ordinates of the empirical periodogram of the series, that is, the reasonable values of `power` need not exclude a sizable fraction of the original sample size. The estimator also allows for the removal of one or more initial ordinates and for the averaging of the periodogram over adjacent frequencies. The rationale for using non-default values of either of these options is presented in Robinson (1995).

Robinson (1995) proposes an alternative log-periodogram regression estimator which he claims provides "modestly superior asymptotic efficiency to $\bar{d}(0)$", ($\bar{d}(0)$ being the Geweke and Porter-Hudak estimator) Robinson (1995, 1052). Robinson's formulation of the log-periodogram regression also allows for the formulation of a multivariate model, providing justification for tests that different time series share a common differencing parameter. Normality of the underlying time series is assumed, but Robinson claims that other conditions underlying his derivation are milder than those conjectured by GPH.

We present here Robinson's multivariate formulation, which applies to a single time series as well. Let X_t represent a G-dimensional vector with g^{th} element $X_{gt}, g = 1, \ldots, G$. Assume that X_t has a spectral density matrix $\int_{-\pi}^{\pi} e^{ij\lambda} f(\lambda)\, d\lambda$, with (g, h) element denoted as $f_{gh}(\lambda)$. The gth diagonal element, $f_{gg}(\lambda)$, is the power spectral density of X_{gt}. For $0 < C_g < \infty$ and $-1/2 < d_g < 1/2$, assume that $f_{gg}(\lambda) \sim C_g \lambda^{-2d_g}$ as $\lambda \rightarrow 0+$ for $g = 1, \ldots, G$. The periodogram of X_{gt} is then denoted as

$$I_g(\lambda) = (2\pi n)^{-1}\left|\sum_{t=1}^{n} X_{gt} e^{it\lambda}\right|^2, g = 1, \ldots, G \tag{9}$$

Without averaging the periodogram over adjacent frequencies nor omission of l initial frequencies from the regression, we may define $Y_{gk} = \log I_g(\lambda_k)$. The least squares estimates of $c = (c_1, \ldots, c_G)'$ and $d = (d_1, \ldots, d_G)'$ are given by

$$\begin{bmatrix} \tilde{c} \\ \tilde{d} \end{bmatrix} = \operatorname{vec}\left\{Y'Z(Z'Z)^{-1}\right\} \tag{10}$$

where $Z = (Z_1, \ldots, Z_m)'$, $Z_k = (1, -2\log\lambda_k)'$, $Y = (Y_1, \ldots, Y_G)$, and $Y_g = (Y_{g,1}, \ldots, Y_{g,m})'$ for m periodogram ordinates. Standard errors for $\tilde{d}_g$ and for a test of the restriction that two or more of the d_g are equal may be derived from the estimated covariance matrix of the least squares coefficients. The standard errors for the estimated parameters are derived from a pooled estimate of the variance in the multivariate case, so that their interval estimates differ from those of their univariate counterparts. Modifications to this derivation when the frequency-averaging (`j`) or omission of initial frequencies (`l`) options are selected may be found in Robinson (1995).

Examples

Data from Terence Mills' *Econometric Analysis of Financial Time Series* on UK FTA All Share stock returns (`ftaret`) and dividends (`ftadiv`) are analyzed.

```
. use http://fmwww.bc.edu/ec-p/data/Mills2d/fta.dta

. tsset
        time variable:  month, 1965m1 to 1995m12

. gphudak ftaret,power(0.5 0.6 0.7)

GPH estimate of fractional differencing parameter
------------------------------------------------------------------------------
                                                        Asy.
Power   Ords    Est d   StdErr  t(H0: d=0)  P>|t|    StdErr  z(H0: d=0)  P>|z|
------------------------------------------------------------------------------
  .50     20  -.00204  .160313    -0.0127   0.990   .187454    -0.0109   0.991
  .60     35  .228244  .145891     1.5645   0.128   .130206     1.7529   0.080
  .70     64  .141861  .089922     1.5776   0.120   .091267     1.5544   0.120
------------------------------------------------------------------------------

. modlpr ftaret, power(0.5 0.55:0.8)

Modified LPR estimate of fractional differencing parameter
------------------------------------------------------------------------------
Power   Ords      Est d    Std Err  t(H0: d=0)    P>|t|    z(H0: d=1)    P>|z|
------------------------------------------------------------------------------
  .50     19   .0231191    .139872     0.1653     0.870      -6.6401     0.000
  .55     25   .2519889   .1629533     1.5464     0.135      -5.8322     0.000
  .60     34   .2450011   .1359888     1.8016     0.080      -6.8650     0.000
  .65     46   .1024504   .1071614     0.9560     0.344      -9.4928     0.000
  .70     63   .1601207   .0854082     1.8748     0.065     -10.3954     0.000
  .75     84   .1749659     .08113     2.1566     0.034     -11.7915     0.000
  .80    113   .0969439   .0676039     1.4340     0.154     -14.9696     0.000
------------------------------------------------------------------------------

. roblpr ftaret

Robinson estimates of fractional differencing parameter
-----------------------------------------------------
Power   Ords      Est d    Std Err  t(H0: d=0)    P>|t|
-----------------------------------------------------
  .90    205   .1253645   .0446745     2.8062     0.005
-----------------------------------------------------

. roblpr ftap ftadiv

Robinson estimates of fractional differencing parameters
Power =  .90                            Ords        =    205
--------------------------------------------------------------
Variable        |     Est d       Std Err      t       P>|t|
----------------+---------------------------------------------
ftap            |  .8698092      .0163302  53.2640     0.000
ftadiv          |  .8717427      .0163302  53.3824     0.000
--------------------------------------------------------------
Test for equality of d coefficients:   F(1,406) =  .00701   Prob > F = 0.9333

. constraint define 1 ftap=ftadiv

. roblpr ftap ftadiv ftaret, c(1)

Robinson estimates of fractional differencing parameters
Power =  .90                            Ords        =    205
--------------------------------------------------------------
Variable        |     Est d       Std Err      t       P>|t|
----------------+---------------------------------------------
ftap            |  .8707759      .0205143  42.4473     0.000
ftadiv          |  .8707759      .0205143  42.4473     0.000
ftaret          |  .1253645      .0290116   4.3212     0.000
--------------------------------------------------------------
Test for equality of d coefficients:   F(1,610) =  440.11   Prob > F = 0.0000
```

The GPH test, applied to the stock returns series, generates estimates of the long memory parameter that cannot reject the null at the ten percent level using the t test. Phillips' modified LPR, applied to this series, finds that $d = 1$ can be rejected for all powers tested, while $d = 0$ (stationarity) may be rejected at the ten percent level for powers 0.6, 0.7, and 0.75. Robinson's estimate for the returns series alone is quite precise. Robinson's multivariate test, applied to the price and dividends series, finds that each series has $d > 0$. The test that they share the same d cannot be rejected. Accordingly, the test is applied to all three series subject to the constraint that price and dividends series have a common d, yielding a more precise estimate of the difference in d parameters between those series and the stock returns series.

Saved Results

`gphudak` saves in `e()`:

`e(N_powers)`	number of powers (scalar)
`e(depvar)`	dependent variable name (macro)
`e(gph)`	matrix of results, 9 by `N_powers`

`modlpr` saves in `e()`:

`e(N_powers)`	number of powers (scalar)
`e(depvar)`	dependent variable name (macro)
`e(modlpr)`	matrix of results, 8 by `N_powers`

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

`r(N)`	number of observations
`r(rob)`	d estimate
`r(se)`	estimated standard error of d
`r(t)`	t statistic
`r(p)`	p-value of t statistic

If more than one power is specified in `roblpr`, the saved results pertain to the last power used.

Acknowledgments

The first author acknowledges John Barkoulas' original exposition of the ARFIMA model, and thanks Peter Phillips for clarifying comments on his working papers. Any remaining errors are the authors' responsibility.

References

Baillie, R. 1996. Long memory processes and fractional integration in econometrics. *Journal of Econometrics* 73: 5–59.

Geweke, J. and S. Porter-Hudak. 1983. The estimation and application of long memory time series models. *Journal of Time Series Analysis* 4: 221–238.

Granger, C. W. J. and R. Joyeux. 1980. An introduction to long-memory time series models and fractional differencing. *Journal of Time Series Analysis* 1: 15–39.

Hosking, J. R. M. 1981. Fractional differencing. *Biometrika* 68: 165–176.

Phillips, P. C. B. 1999a. Discrete Fourier transforms of fractional processes. Unpublished working paper No. 1243, Cowles Foundation for Research in Economics, Yale University. `http://cowles.econ.yale.edu/P/cd/d12a/d1243.pdf`

——. 1999b. Unit root log periodogram regression. Unpublished working paper No. 1244, Cowles Foundation for Research in Economics, Yale University. `http://cowles.econ.yale.edu/P/cd/d12a/d1244.pdf`

Robinson, P. M. 1995. Log-periodogram regression of time series with long range dependence. *Annals of Statistics* 23: 1048–1072.

Sowell, F. 1992. Maximum likelihood estimation of stationary univariate fractionally-integrated time-series models, *Journal of Econometrics* 53: 165–188.

sts17 | Compacting time series data

Christopher F. Baum, Boston College, baum@bc.edu

Abstract: `tscollap` provides the ability to compact data of monthly, quarterly or half-yearly frequencies to a lower frequency by one or more methods (e.g., average, sum, last value per period, and so on).

Keywords: time series, data frequency, collapse.

Syntax

`tscollap` *clist* `,` `to(`*freq*`)` [`generate(`*freqvar*`)`]

where *clist* is either

[`(`*stat*`)`] *varlist* [[`(`*stat*`)`] ...]

or

[`(`*stat*`)` *target_var* `=` *varname* [*target_var* `=` *varname* ...] [[`(`*stat*`)` ...]]

or any combination of the *varlist* or *target_var* forms, and *stat* is one of

`mean`	mean over interval
`sum`	sum over interval
`gmean`	geometric mean over interval (for a positive variable)
`first`	first observation in the interval
`last`	last observation in the interval

If *stat* is not specified, `mean` is assumed.

The `tscollap` command is for use with time series data of monthly, quarterly, or half-yearly frequency. You must `tsset` your data before using this command; see [U] **tsset**. If the data are a panel of time series, that is, if a *panelvar* has been specified in `tsset`, the specification of the panel identification variable will automatically be retained in the resulting dataset (it need not, and should not, be specified in the *varlist*). Time series operators may not be used.

The `tsmktim` command (described in *dm81*; see pages 2–4) is a convenient way to generate the appropriate `tsset` command if you do not already have a time variable in the data.

Options

`to(`*freq*`)` specifies the target frequency, which must be specified. It may take on any value lower than the current value as understood by `tsset`. *freq* must be given as `q, h, y` in either lowercase or uppercase.

`generate(`*freqvar*`)` may be used to specify the name of the new `tsset` variable, which will be formatted at the target frequency.

Description

`tscollap` converts the time series data in memory into a dataset of means, sums, or selected values taken from the specified interval. It is a variant of `collapse`, which automatically forms the groups over which statistics are to be calculated from an understanding of the calendar data. For instance, monthly data may be converted to quarterly, half-yearly, or annual (yearly) data by specifying `to(q), to(h), or to(y)`, respectively. Data may be averaged over the interval (using either an arithmetic or geometric mean) or summed (as would be appropriate for income statement data). Either the first or the last observation of each interval may be selected (so that, e.g., end-of-period values may be readily assembled). Since its syntax (and internal logic) is taken from `collapse`, more than one statistic may be generated from a single variable; for example, both average and end-of-period values may be specified by using different *target_var* names). `tscollap` embodies the June, 2000 correction to `collapse`.

All variables not specified in the target list are dropped (including the current `tsset` variable), and a new `tsset` variable is generated as *freq_freq* (as long as that variable does not already exist). The `generate` option may be used to customize the new `tsset` variable. If a *panelid* variable is in use by `tsset`, it should not be listed; it will be automatically retained.

Saved results

`tscollap` saves the items returned by `tsset` in `r()`.

Remarks

`tscollap` makes substantial use of `_gfilter`, part of the `egenmore` package of N. J. Cox, information about which is available by issuing the command `webseek egenmore`.

Examples

Monthly data from Terence Mills' *Econometric Analysis of Financial Time Series* on UK FTA All Share stock prices (`ftap`) and dividends (`ftadiv`) are compacted to a quarterly frequency.

```
. use http://fmwww.bc.edu/ec-p/data/Mills2d/fta.dta
. tscollap ftap ftadiv, to(q)
Converting from M to Q
        time variable:  q_q, 1965q1 to 1995q4
. use http://fmwww.bc.edu/ec-p/data/Mills2d/fta.dta
. tscollap ftap (first) ftapf=ftap (last) ftapl=ftap, to(q) gen(qtr)
Converting from M to Q
        time variable:  qtr, 1965q1 to 1995q4
```

In the first instance, the price and dividend series are averaged over the quarter, and other series in the original dataset discarded. In the second example, the price series is used to generate three variables: the average price per quarter, the price in the first month of each quarter, and the price in the last month of each quarter, as `ftap`, `ftapf`, and `ftapl`, respectively.

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	*qs*	questions and suggestions
dt	datasets	*tt*	teaching
gr	graphics	*zz*	not elsewhere classified
in	instruction		

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
Address:	P.O. Box 1169 17100 NAZERATH-ELLIT Israel
Phone:	+972 (0)6 6100101
Fax:	+972 (0)6 6554254
Email:	assc@netvision.net.il
Countries served:	Israel

Company:	Axon Technology Company Ltd
Address:	9F, No. 259, Sec. 2 Ho-Ping East Road TAIPEI 106 Taiwan
Phone:	+886-(0)2-27045535
Fax:	+886-(0)2-27541785
Email:	hank@axon.axon.com.tw
Countries served:	Taiwan

Company:	Chips Electronics
Address:	Lokasari Plaza 1st Floor Room 82 Jalan Mangga Besar Raya No. 81 JAKARTA Indonesia
Phone:	62 - 21 - 600 66 47
Fax:	62 - 21 - 600 66 47
Email:	puyuh23@indo.net.id
Countries served:	Indonesia, Brunei, Malaysia Singapore

Company:	Dittrich & Partner Consulting
Address:	Kieler Strasse 17 5. floor D-42697 Solingen Germany
Phone:	+49 2 12 / 26 066 - 0
Fax:	+49 2 12 / 26 066 - 66
Email:	sales@dpc.de
URL:	http://www.dpc.de
Countries served:	Germany, Austria, Czech Republic Hungary, Italy, Poland

Company:	IEM
Address:	P.O. Box 2222 PRIMROSE 1416 South Africa
Phone:	+27-11-8286169
Fax:	+27-11-8221377
Email:	iem@hot.co.za
Countries served:	South Africa, Botswana, Lesotho, Namibia, Mozambique, Swaziland, Zimbabwe

Company:	MercoStat Consultores
Address:	9 de junio 1389 CP 11400 MONTEVIDEO Uruguay
Phone:	598-2-613-7905
Fax:	Same
Email:	mercost@adinet.com.uy
Countries served:	Uruguay, Argentina, Brazil, Paraguay

Company:	Metrika Consulting
Address:	Mosstorpsvagen 48 183 30 Taby STOCKHOLM Sweden
Phone:	+46-708-163128
Fax:	+46-8-7924747
Email:	sales@metrika.se
URL:	http://www.metrika.se
Countries served:	Sweden, Baltic States, Denmark, Finland, Iceland, Norway

Company:	Ritme Informatique
Address:	34, boulevard Haussmann 75009 Paris France
Phone:	+33 (0)1 42 46 00 42 +33 (0)1 42 46 00 33
Email:	info@ritme.com
URL:	http://www.ritme.com
Countries served:	France, Belgium, Luxembourg

(*List continued on next page*)

International Stata Distributors

(*Continued from previous page*)

Company:	Scientific Solutions S.A.
Address:	Avenue du Général Guisan, 5 CH-1009 Pully/Lausanne Switzerland
Phone:	41 (0)21 711 15 20
Fax:	41 (0)21 711 15 21
Email:	info@scientific-solutions.ch
Countries served:	Switzerland

Company:	Smit Consult
Address:	Doormanstraat 19 5151 GM Drunen Netherlands
Phone:	+31 416-378 125
Fax:	+31 416-378 385
Email:	info@smitconsult.nl
URL:	http://www.smitconsult.nl
Countries served:	Netherlands

Company:	Survey Design & Analysis Services Pty Ltd
Address:	PO Box 1026 Blackburn North VIC 3130 Australia
Phone:	+61 (0)3 9878 7373
Fax:	+61 (0)3 9878 2345
Email:	sales@survey-design.com.au
URL:	http://survey-design.com.au
Countries served:	Australia, New Zealand

Company:	Timberlake Consultants
Address:	Unit B3 Broomsleigh Bus. Park Worsley Bridge Road LONDON SE26 5BN United Kingdom
Phone:	+44 (0)208 697 3377
Fax:	+44 (0)208 697 3388
Email:	info@timberlake.co.uk
URL:	http://www.timberlake.co.uk
Countries served:	United Kingdom, Eire

Company:	Timberlake Consultants Srl
Address:	Via Baden Powell, 8 67039 SULMONA (AQ) Italy
Phone:	+39 (0)864 210101
Fax:	+39 (0)864 32939
Email:	timberlake@arc.it
URL:	http://www.timberlake.it
Countries served:	Italy

Company:	Timberlake Consulting S.L.
Address:	Calle Mendez Nunez, 1, 3 41011 Sevilla Spain
Phone:	+34 (9) 5 422 0648
Fax:	+34 (9) 5 422 0648
Email:	timberlake@zoom.es
Countries served:	Spain

Company:	Timberlake Consultores, Lda.
Address:	Praceta Raúl Brandao, n°1, 1°E 2720 ALFRAGIDE Portugal
Phone:	+351 (0)1 471 73 47
Fax:	+351 (0)1 471 73 47
Email:	timberlake.co@mail.telepac.pt
Countries served:	Portugal

Company:	Unidost A.S. Rihtim Cad. Polat Han D:38 Kadikoy 81320 ISTANBUL Turkey
Phone:	+90 (216) 414 19 58
Fax:	+30 (216) 336 89 23
Email:	info@unidost.com
URL:	http://abone.turk.net/unidost
Countries served:	Turkey

Company:	Vishvas Marketing-Mix Services
Address:	C\O S. D. Wamorkar "Prashant" Vishnu Nagar, Naupada THANE - 400602 India
Phone:	+91-251-440087
Fax:	+91-22-5378552
Email:	vishvas@vsnl.com
Countries served:	India